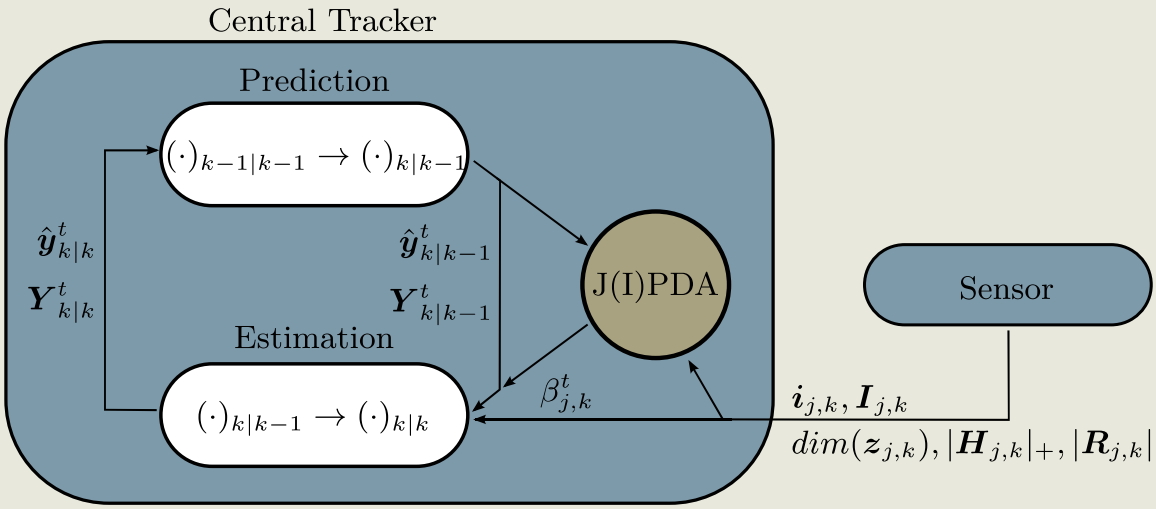
ulm university universität ulm



Generic Sensor Data Fusion in Information Space and a New Approach to Processing Dense Sensor Data

Benjamin Wilking

Band 25

Schriftenreihe des Instituts für Mess-, Regel- und Mikrotechnik

Benjamin Wilking

Generic Sensor Data Fusion in Information Space and a New Approach to Processing Dense Sensor Data

Schriftenreihe des
Instituts für Mess-, Regel- und Mikrotechnik
Universität Ulm

Herausgeber:
Prof. Dr.-Ing. Klaus Dietmayer

Band 25

Benjamin Wilking

Generic Sensor Data Fusion in Information Space and a New Approach to Processing Dense Sensor Data

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Dissertation, Universität Ulm,
Fakultät für Ingenieurwissenschaften, Informatik und Psychologie, 2017

Impressum

Universität Ulm
Institut für Mess-, Regel- und Mikrotechnik
Prof. Dr.-Ing. Klaus Dietmayer
Albert-Einstein-Allee 41
89081 Ulm
http://www.uni-ulm.de/mrm

Eine Übersicht über alle Bände der Schriftenreihe finden Sie unter
http://www.uni-ulm.de/mrmschriften.

Diese Veröffentlichung ist im Internet auf dem Open Access Repositorium der Universität Ulm (https://oparu.uni-ulm.de) verfügbar und dort unter der Lizenz “Standard ohne Print-On-Demand” publiziert.
Details zur Lizenz sind unter http://www.uni-ulm.de/index.php?id=50392 zu finden.

Institut für Mess-, Regel- und Mikrotechnik der Universität Ulm 2017
Print on Demand

ISBN 978-3-941543-37-9 e-ISBN 978-3-941543-38-6

ulm university universität
uulm

Generic Sensor Data Fusion in Information Space and a New Approach to Processing Dense Sensor Data

DISSERTATION

zur Erlangung des akademischen Grades eines

DOKTOR-INGENIEURS
(Dr.-Ing.)

der Fakultät für Ingenieurwissenschaften, Informatik
und Psychologie der Universität Ulm

von

Benjamin Wilking
aus Laupheim

Gutachter:	Prof. Dr.-Ing. Klaus Dietmayer
	Prof. Dr.-Ing. Gerd Wanielik
Amtierender Dekan:	Prof. Dr. rer. nat. Frank Kargl

Ulm, 24.11.2017

Danksagung

Diese Arbeit ist das Ergebnis meiner Zeit am Institut für Mess-, Regel- und Mikrotechnik der Universität Ulm. Es waren für mich spannende Jahre, in denen ich viele interessante Dinge und noch mehr nette Menschen kennen lernen durfte. Der tägliche Weg ins Büro fiel Dank des wundervollen Umgangs miteinander auch in schwierigeren Zeit stets leicht. Ich möchte mich daher bei meinem Betreuer Prof. Klaus Dietmayer dafür bedanken, dass er dieses Umfeld stets unterstützt und mir diese Arbeit ermöglich hat. Bei Claudia Fricke bedanke ich mich dafür, dass sie stets die gute Seele des Instituts ist und bei Michael Buchholz und Jürgen Remmlinger für die LaTeX-Vorlage: die gewonnene Zeit ist unbezahlbar. Bei Thorsten Weiß bedanke ich mich, weil er mich schon als Student zu einer Promotion am Institut animiert hat: ohne ihn hätte ich mich wohl nicht zu diesem Schritt entschieden. Vielen Dank auch an Prof. Gerd Wanielik für die Übernahme des Zweitgutachtens.
Eine Aufzählung all der Kollegen, die ich an dieser Stelle gerne erwähnen würde, wäre wohl zu viel des Guten. Einigen jedoch möchte ich ganz besonders danken: Allen voran Stephan Reuter, mit dem ich mir lange Zeit das Büro geteilt habe und der mir vom ersten Tag an mit Rat und Tat zur Seite stand. Meine fachlichen Diskussionen mit ihm waren stets wertvoll und unsere freundschaftlichen Wortgefechte waren eine willkommene Ablenkung. Daniel Meißner, Elias Strigel und Florian Seeliger für unsere gemeinsame Zeit, die ich nicht missen möchte: unsere nächtlichen Projekttreffen in Aschaffenburg sind unvergesslich. Hendrik Deusch, Jürgen Wiest, Dominik Nuß und Felix Kunz für die vielen Diskussionen und Freitagnachmittage.
Vielen Dank euch allen dafür, dass wir zusammen etwas bewegt haben.

Ein großes Dankeschön geht an meine Mutter und meine Schwiegereltern: ohne die tatkräftige Unterstützung bei der Kinderbetreuung wäre diese Arbeit vielleicht nicht mehr beendet worden.

Zu guter Letzt und doch allen voran danke ich meiner Frau Svenja. Für ihre Unterstützung, das viele Zuhören und die Aufmunterung. Dafür, dass sie mich seit vielen Jahren auf allen Wegen begleitet und mit mir beruflich wie auch privat die höchsten Höhen und tiefsten Tiefen durchwandert. Nach all den erlebten Abenteuern freue ich mich darauf, noch viele Wege mit ihr zu gehen.

Abstract

Sensor data fusion is the key to a comprehensive environment perception by today's and future systems basing on object tracking. Early and rather simple advanced driver assistance systems (ADAS) are still using a sensor setup with a single sensor where the problem of fusing different data sources does not arise. With the intent on fusing data from multiple heterogeneous sensors into one common object tracking system, various fusion methods are conceivable. Probabilistic data association (PDA) is one of these methods and it has been shown to be feasible and effective in former publications [Mäh09; Mun11]. In very complex scenarios, algorithms based on finite set statistics [Mah03] have become popular over the last few years. These allow the modeling of interactions between the tracked objects to resolve ambiguities comparable to the multi-object Bayes filter. Independent of the fusion method, all procedures share one major drawback: the interchangeability of the sensors is not possible. In many systems it is necessary to transmit knowledge about the sensors to the fusion system. Thus, a change of the sensor setup can entail comprehensive consequences and can be expensive and costly in development time. Besides increasing the sensor interchangeability, the anonymization of the sensor is desirable. This allows sensors to be used without knowledge of the sensors' theoretical principles and enables the sensor manufacturer to maintain secrecy about the details. The generic linkage of sensors to object tracking systems as well as the anonymization of sensors is the purpose of this work. Therefore, a mathematically equivalent alternative to the Kalman filter, the information filter, is used. The focus is set on probabilistic data association and the successful adaption of it to use the information filter is evaluated in simulation and real-data scenarios. Additionally, it is shown how to use the information filter approach in many other fusion systems.

In the further course of this work a novel approach to preprocessing high density data from distance measuring sensors is presented. This new approach meets the requirements on generically linked sensors. It allows the use of the information space and simultaneously increases the perception performance markedly in comparison to former attempts. This is achieved by filtering the raw sensor data over time and generating reliable object hypotheses using the filtered data. The performance of the achieved sensor model is demonstrated in various real-data scenarios.

Kurzfassung

Sensordatenfusion ist der Schlüssel zu heutigen und zukünftigen Systemen, die auf einer Objektverfolgung zur umfassenden Umgebungserfassung beruhen. Frühe und einfache Systeme basieren noch heute auf lediglich einem Sensor, wodurch sich das Problem der Fusion verschiedener Datenquellen nicht ergibt. Sollen jedoch Daten mehrerer heterogener Sensoren in eine gemeinsame Objektverfolgung eingebracht werden, kommen diverse Fusionsmethoden in Frage. Die probabilistische Datenassoziation (PDA) ist eine dieser Methoden und hat sich in Untersuchungen früherer Publikationen [Mäh09; Mun11] als einfach und wirkungsvoll erwiesen. In sehr komplexen Szenarien kommen in den letzten Jahren auch vermehrt Methoden der Statistik endlicher Mengen (engl. finite set statistics (FISST)) [Mah03] zum Einsatz. Diese erlauben zusätzlich die Modellierung von Abhängigkeiten zwischen den Objekten, wodurch Mehrdeutigkeiten aufgelöst werden können. Unabhängig von der Fusionsmethode teilen sich alle Verfahren jedoch ein Problem: die Austauschbarkeit der Sensorik ist nicht vollständig gewährleistet. Es muss stets Wissen über die verwendeten Sensoren in das System eingebracht werden. Eine Veränderung des Sensoraufbaus kann weitreichende Konsequenzen nach sich ziehen und verursacht sowohl Zeit- als auch Kostenaufwand. Des Weiteren ist neben dem Vereinfachen des Sensoraustauschs auch eine Anonymisierung des Sensors wünschenswert. Dies würde es erlauben, Sensoren ohne Wissen über ihr Messprinzip zu verwenden. Auf der anderen Seite müsste der Hersteller dann keine detaillierten Informationen über den Sensor preisgeben. Die generische Anbindung von Sensoren an fusionierende Systeme zur Objektverfolgung sowie die Anonymisierung der Sensoren sind Ziel dieser Arbeit. Um dies zu realisieren, wird der Informationsfilter als Alternative zum mathematisch äquivalenten Kalman Filter verwendet. Besonderes Augenmerk wird dabei auf die PDA gelegt. Deren erfolgreiche Anpassung wird anhand von Simulation und realer Daten ausgewertet. Darüber hinaus wird gezeigt, wie der Informationsfilter-Ansatz auch in vielen anderen Fusionsmethoden eingesetzt werden kann.
Im weiteren Verlauf dieser Arbeit wird ein neuartiger Ansatz zur Vorverarbeitung für hochauflösende und Distanz messende Sensoren präsentiert, die den Anforderungen an generisch angebundenen Sensoren entspricht. Dies ermöglicht den Einsatz des Informationsraumes und verbssert gleichzeitig die Detektionsleistung deutlich gegenüber bisherigen Ansätzen. Dabei werden direkt die Sensorrohdaten zeitlich verfolgt, um zuverlässige Hypothesen erzeugen zu können. Die Leistungsfähigkeit des neuen Ansatzes wird in verschiedenen realen Szenarien nachgewiesen.

Contents

Chapter 1

Introduction

Nearly all kinds of modern object tracking systems are getting more and more complex to improve existing systems or to allow the development of new ones. Therefore, the amount of simultaneously used sensors increases in order to get more precise information about the environment and thus a highly detailed environment perception. In the field of ADAS, especially the intensely investigated highly autonomous driving, data of multiple sensors has to be combined [KNW+15]. As soon as there are several sensors, mostly of different kinds, a fusion system is necessary. Without a dedicated fusion system, the information of different sources is combined in an heuristic manner and leads to a highly sensor dependent and static setup. Fusion systems allow to model an interface in a generic way. Today, there are fusion systems which can treat measurements generically, but most of them still share the same problem: to interpret measurements, they depend on knowledge about the connected devices and, not seldom, on heuristics. Every change in the sensor setup entails a major change of the fusion system or even a complete new development cycle, which causes high costs and takes a lot of time. Therefore, a generic sensor data fusion, which allows to regard sensors as "plug & play" devices, would be extremely useful. Munz et al. showed in [MDM09] that the joint integrated probabilistic data association (JIPDA) [ME02] is an appropriate algorithm for a generic fusion framework. The JIPDA allows the fusion of sensor data and track management in a probabilistic manner and can handle information from different sources. Nevertheless, there are still some parts in the algorithm for which knowledge about the sensor devices is necessary, since not every sensor provides the same type of measurements. These depend on the measurement principle and may even vary over time.

To the knowledge of the author, the first part of this work is the first contribution presenting a solution to the just described problem by combining the JIPDA with the information filter and solving some of the mathematical problems coming along with the integration of the information filter into the JIPDA algorithm. The information filter, as presented in [GDH92], is equivalent to the Kalman filter and is often used for

reasons of less computational effort or better decentralization characteristics [Mut98]. Here, the focus is not on reducing the computational load, but on gaining advantages in generic data fusion by combining the information filter with the probabilistic data association approach. One major advantage in using the information filter is a common interface for all kinds of sensors. Therefore, the amount of data to transmit in every cycle is always the same and known in advance. This has the effect, that all information about the sensor and its measurement principle is encapsulated in the measurement and can not be recovered in detail - which may be seen as a great advantage by sensor manufacturers.

Independent from the fusion system, the quality of a sensor setup is of course a matter of the quality of the sensor data. In case of single-measurement sensors, there are only few possibilities to influence the perception result. On the opposite, when using high density distance measuring sensors, further called HDDM, the perception strongly depends on the preprocessing. Such sensors provide dense or semi dense point clouds instead of single-measurements. Thus, the task of the preprocessing is creating measurement hypotheses from these point clouds. Therefore, a new method, where the raw data is filtered over time, is presented in the second part of this work. The filter process extends frame based clustering approaches and allows the use of information like velocity and yaw angle. The main advantages of the method presented here are the improvement of the perception range, the detection rate, and the independence of any model assumption. This is evaluated in several real-data scenarios, where vehicles, pedestrians, and bicycles are observed in different surroundings.

Chapter 2

Kalman Filter based Data Fusion

The cornerstone of nearly every state of the art sensor data fusion system is the widely spread Kalman filter [Kal60]. Today, the mostly used approach to filtering is Bayesian. This was presented in [HL64] as an extension to the former least squares version of Kalman. Filtering an object's state over time is the recursive estimation of the state by incorporating knowledge about the process model, e.g. the movement, of the object and an uncertain measurement about the object's state. As Ho et al. showed, using only Gaussian distributions and linear models, the Kalman filter is a Bayes optimal estimator [HL64]. Later, Grime et al. introduced in [GDH92] the so called information filter. Mathematically equivalent to the Kalman filter, the information filter has some advantages in the decentralized fusion of information, since the information contribution of different sources can be fused by simply summing them up. In this work it is shown, that the information filter approach has additional advantages in fusing measurement data from different sensors about the same objects and therefore has some advantages for generic sensor data fusion which are mentioned above.

This chapter briefly reviews the basics of the Kalman and the information filter. Afterwards, the state of the art fusion systems using soft and hard decision association are introduced. In such systems, one tracker is instantiated for every perceived object and thus relies on a sufficient data association method. The probabilistic data association (PDA) is one of these algorithms and is the focus of this work. The chapter closes with the introduction of performance evaluation metrics for multi-object tracking.

2.1 Kalman Filter

The Kalman filter and its equations as introduced in [HL64; Kal60] are an important part of this work. Thus, a brief review is given in this section. The idea of the Kalman filter is to observe the state of a system over time using a series of measurements. In order to do so, knowledge about the observed system is necessary. In case of tracking applications, this knowledge inherits information about the behavior of the tracked object. In this work, the object behavior is represented using a process model to predict the movement of the object. This movement is described by the linear model equation:

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}_k \boldsymbol{x}_k + \boldsymbol{G}_k \boldsymbol{u}_k + \boldsymbol{v}_k. \tag{2.1}$$

Therein $\boldsymbol{x}_k$ is the state vector and represents the state of the object in state space at time step k. $\boldsymbol{F}_k$ is the state transition matrix and contains the process model. The further parts of the equation are the control input matrix $\boldsymbol{G}_k$ and the control vector $\boldsymbol{u}_k$. The vector $\boldsymbol{v}_k$ is a discrete zero-mean white Gaussian noise process and allows to incorporate uncertainties about the underlying model. With that process noise, the process noise matrix can be expressed as the expectation:

$$\boldsymbol{Q}_k = \mathrm{E}\left\{\boldsymbol{v}_k \boldsymbol{v}_k^{\mathrm{T}}\right\}. \tag{2.2}$$

The state of the object can be observed using a sensor generating a measurement:

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{w}_k. \tag{2.3}$$

The measurement matrix $\boldsymbol{H}_k$ describes the sensor model and the vector $\boldsymbol{w}_k$ is a zero-mean white Gaussian noise, representing the measurement noise and the uncertainty of the measurement model. The measurement covariance matrix is defined as the expectation:

$$\boldsymbol{R}_k = \mathrm{E}\left\{\boldsymbol{w}_k \boldsymbol{w}_k^{\mathrm{T}}\right\}. \tag{2.4}$$

The Kalman filter is composed of two recursive parts: first, the estimated state $\hat{\boldsymbol{x}}_{k-1|k-1}$ of the last time step and its covariance $\boldsymbol{P}_{k-1|k-1}$ are predicted over time. In the second step, the state and the covariance are updated to obtain a new estimation using the measurement $\boldsymbol{z}_k$. These two steps are called *prediction* and *estimation* (see Figure 2.1). The estimation step is also known as: innovation step, update step, or corrector step.

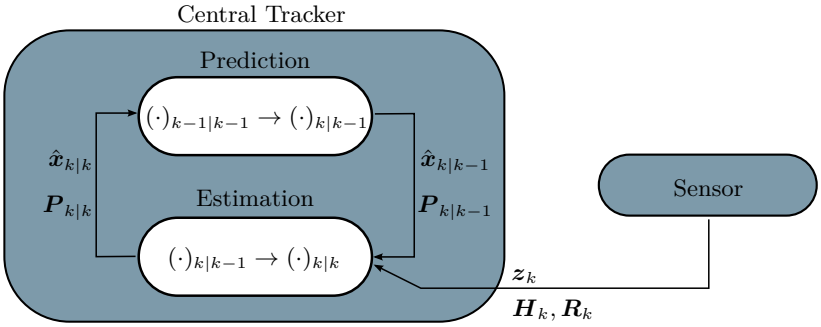


Figure 2.1: Single-object Kalman filter in state space.

The prediction equations of the state and the covariance are:

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}_{k-1}\hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{G}_{k-1}\boldsymbol{u}_{k-1} \tag{2.5}$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1|k-1}\boldsymbol{F}_{k-1}^{\mathrm{T}} + \boldsymbol{Q}_{k-1}. \tag{2.6}$$

The estimation equations are given with:

$$\hat{\boldsymbol{z}}_{k|k-1} = \boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k-1} \tag{2.7}$$

$$\boldsymbol{S}_k = \boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^{\mathrm{T}} + \boldsymbol{R}_k \tag{2.8}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{S}_k^{-1} \tag{2.9}$$

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k\left[\boldsymbol{z}_k - \hat{\boldsymbol{z}}_{k|k-1}\right] \tag{2.10}$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_k\boldsymbol{S}_k\boldsymbol{K}_k^{\mathrm{T}}. \tag{2.11}$$

Therein $\hat{\boldsymbol{z}}_{k|k-1}$ is the predicted measurement under the assumption of the predicted state $\hat{\boldsymbol{x}}_{k|k-1}$. The matrix $\boldsymbol{S}_k$ is the innovation covariance and the matrix $\boldsymbol{K}_k$ is known as the Kalman gain.

So far, the process model in Equation (2.5) and the measurement model in Equation (2.7) assume linear functions. In case of non-linear functions the Kalman filter equations as given above need to be adapted. There are several approximations to handle these non-linearities. The most commonly used are the extended Kalman filter (EKF) [BL98] and the unscented Kalman filter (UKF) [JUD00].

2.2 Information Filter

The information filter is also known as the *inverse covariance* form of the Kalman filter [BL98]. It is mathematically equivalent to the Kalman filter but uses measures of information about the states, instead of the states and the covariances itself [GDH92; Mut98]. As described in [Dur01], information is a measure of the compactness of a distribution. This means that, if a probability distribution is widely spread, the amount of information is very low. If there are clear peaks, the information content is high. Thus, information is a function of the distribution. In this section, the information space and the corresponding information filter are introduced and compared with the standard Kalman filter. The explanations and equations of this section are taken from [Mut98].

2.2.1 Information Space

Bayesian Theory

The Kalman filter, and therefore the information filter as well, are basing on the Bayes theorem. This theorem allows to estimate $\boldsymbol{x}$ by the knowledge contained in $\boldsymbol{z}$ using the conditional probability distribution function $p(\boldsymbol{z}|\boldsymbol{x})$, also known as the *likelihood function*. Following the likelihood principle, it is assumed, that the likelihood function inherits all necessary information about $\boldsymbol{x}$ to generate an estimate. In combination with the a priori information $p(\boldsymbol{x})$ about $\boldsymbol{x}$ and $p(\boldsymbol{z})$ about $\boldsymbol{z}$, the Bayes theorem gives the posterior conditional distribution of $\boldsymbol{x}$ given $\boldsymbol{z}$:

$$p(\boldsymbol{x}|\boldsymbol{z}) = \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{z})} \tag{2.12}$$

Taking only one measurement into account normally results in a rather high uncertainty in the estimate of $\boldsymbol{x}$. To reduce this uncertainty, it is possible to use a set Z_k of all measurements of $\boldsymbol{x}$ up to time k:

$$\boldsymbol{Z}_k \triangleq \{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \ldots, \boldsymbol{z}_k\}. \tag{2.13}$$

Thus, the definition of the likelihood has to be changed to handle the set of all measurements. This results in the likelihood function

$$\Lambda_k(\boldsymbol{x}) = p(\boldsymbol{Z}_k|x). \tag{2.14}$$

and therefore in the new a posteriori

$$p(\boldsymbol{x}|\boldsymbol{Z}_k) = \frac{p(\boldsymbol{Z}_k|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{Z}_k)} \tag{2.15}$$

In order to use this equation in a recursive estimator, it can be rewritten as:

$$p(\boldsymbol{x}|\boldsymbol{Z}_k) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{Z}_{k-1})}{p(\boldsymbol{Z}_{k-1})} \tag{2.16}$$

This recursive formulation uses only the measurement of the current time $\boldsymbol{z}_k$. Thus, it is not necessary to store all previous measurements. The recursive form of the Bayes theorem is the most common approach to filtering approaches.

Fisher Information

The Fisher information is a measure of information an observable random variable $\boldsymbol{Z}_k$ contains about a random variable $\boldsymbol{x}$. As depicted in Equation (2.14) the probability function for $p(\boldsymbol{Z}_k|\boldsymbol{x})$ is the likelihood function $\Lambda_k(\boldsymbol{x})$ for $\boldsymbol{x}$. The derivation of the natural logarithm of the likelihood is called the score function $\boldsymbol{s}_k(\boldsymbol{x})$ with

$$\boldsymbol{s}_k(\boldsymbol{x}) \triangleq \nabla_{\boldsymbol{x}} \ln p(\boldsymbol{Z}_k|\boldsymbol{x}). \tag{2.17}$$

Considering the score function as a random variable the expectation equals zero

$$\mathrm{E}\left[\boldsymbol{s}_k\left(\boldsymbol{x}\right)\right] = 0 \tag{2.18}$$

and the Fisher information results in:

$$\boldsymbol{\mathcal{I}}_k = -\mathrm{E}\left[\nabla_{\boldsymbol{x}} \nabla_{\boldsymbol{x}}^{\mathrm{T}} \ln p\left(\boldsymbol{Z}_k|\boldsymbol{x}\right)\right]. \tag{2.19}$$

Another information measure known from estimation and control is the Cramer-Rao lower bound (CRLB) [BL93]. For any unbiased estimator for the state vector $\boldsymbol{x}_k$ the CRLB is the lower bound of the mean squared error vector:

$$\mathrm{E}\left[\left\{\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k}\right\}\left\{\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k}\right\}^{\mathrm{T}}|Z_k\right] \geq \boldsymbol{\mathcal{I}}_k^{-1}. \tag{2.20}$$

Thus, the covariance matrix of the estimator has a lower bound. Equation (2.20) shows, that the CRLB can be expressed as the inverse of the Fisher information matrix. If the likelihood function $p\left(Z_k|\boldsymbol{x}\right)$ is assumed to be Gaussian, by considering the probability distribution of a the random vector $\boldsymbol{x}_k$ as Gaussian with mean $\hat{\boldsymbol{x}}_{k|k}$ and covariance $\boldsymbol{P}_{k|k}$, it can be shown, that the Fisher information matrix is the

inverse of the covariance matrix:

$$\boldsymbol{\mathcal{I}}_k = \boldsymbol{P}_{k|k}^{-1} = (CRLB)^{-1}. \tag{2.21}$$

More details about the Fisher information and its connection to the entropy (Shannon information) can be found in [Mut98] and [Dur01].

2.2.2 Filter Equations

The filter equations of the prediction step in information space are pretty much the same as in state space. Using the equivalences in [Mut98]

$$\boldsymbol{Y}_{k|k}^{-1} = \boldsymbol{P}_{k|k} = \boldsymbol{\mathcal{I}}_k^{-1}, \tag{2.22}$$

$$\hat{\boldsymbol{x}}_{k|k} = \boldsymbol{P}_{k|k}\hat{\boldsymbol{y}}_{k|k}, \tag{2.23}$$

$$\hat{\boldsymbol{y}}_{k|k} = \boldsymbol{Y}_{k|k}\hat{\boldsymbol{x}}_{k|k}, \tag{2.24}$$

where $\boldsymbol{y}_{k|k}$ is the estimated information vector and $\boldsymbol{Y}_{k|k}$ the estimated information matrix, the prediction Equations (2.5) and (2.6) can be written as:

$$\boldsymbol{Y}_{k|k-1} = \left[\boldsymbol{F}_{k-1}\boldsymbol{Y}_{k-1|k-1}^{-1}\boldsymbol{F}_{k-1}^{\mathrm{T}} + \boldsymbol{Q}_{k-1}\right]^{-1} \tag{2.25}$$

$$\hat{\boldsymbol{y}}_{k|k-1} = \boldsymbol{Y}_{k|k-1}\boldsymbol{F}_{k-1}\boldsymbol{Y}_{k-1|k-1}^{-1}\hat{\boldsymbol{y}}_{k-1|k-1}. \tag{2.26}$$

The estimation equations are different to those in state space. As described in [Mut98] the information space version results after some conversions in:

$$\hat{\boldsymbol{y}}_{k|k} = \hat{\boldsymbol{y}}_{k|k-1} + \boldsymbol{i}_k \tag{2.27}$$

$$\boldsymbol{Y}_{k|k} = \boldsymbol{Y}_{k|k-1} + \boldsymbol{I}_k, \tag{2.28}$$

with

$$\boldsymbol{i}_k = \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{R}_k^{-1}\boldsymbol{z}_k \tag{2.29}$$

$$\boldsymbol{I}_k = \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{R}_k^{-1}\boldsymbol{H}_k. \tag{2.30}$$

Therein $\boldsymbol{i}_k$ is called the information gain and $\boldsymbol{I}_k$ the information gain matrix. Together, $\boldsymbol{i}_k$ and $\boldsymbol{I}_k$ are called *information measurement*.
For non-linear systems there are the extended information filter (EIF)[Mut98] and the unscented information filter (UIF)[Mut98]. Both are very similar to the corresponding EKF and UKF.

2.2.3 Comparison of Kalman and Information Filter

The Kalman and the information filter are, as already pointed out, mathematically equivalent. After every prediction and every estimation, a conversion between the two representations is possible using Equations (2.22) to (2.24). Taking a closer look and comparing the Figures 2.1 and 2.2 it is obvious that the information filter provides a completely generic interface for sensors in the single-object case since there is no need to transmit the measurement model or the measurement noise. All the necessary information is incorporated in the information contribution using Equations (2.29) and (2.30).

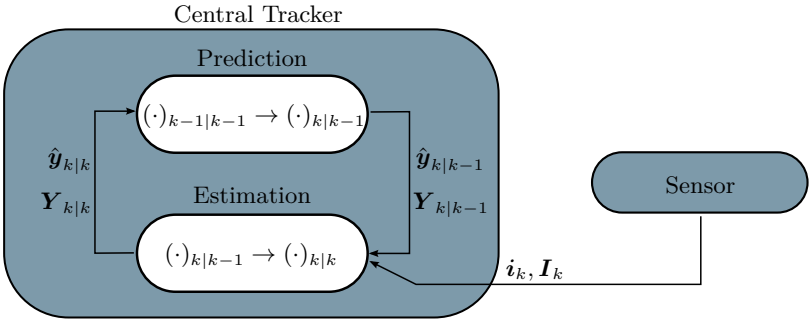


Figure 2.2: Single-object Kalman filter in information space

2.3 Data Association in Fusion Systems

The strongest restriction when tracking one or multiple objects with a standard Kalman filter is the assumption that the sensor provides only one measurement per object. In case of filtering one dimensional sensor data, this assumption may be correct, but for object tracking the used sensors normally are able to provide multiple measurements for one or more objects at the same time. Furthermore, false alarms are usually unavoidable. Considering multiple measurements originated by at least one object, especially designed filter algorithms are used. In case of fusion systems basing on data association, one Kalman filter is instantiated for every perceived object. Thus, an assignment of measurements to tracks is necessary. Such fusion systems are also called multi-instance filters. Since every instantiated tracker needs knowledge about the measurement in the estimation step, the assignment problem is crucial. This association can be done using hard or a soft decision algorithms. For

soft decision the probabilistic data association (PDA) is a common approach. The hard and the probabilistic data association are summed up in this section.

2.3.1 Hard Decision Association

The hard decision association methods are normally realized using nearest neighbor (NN) methods. This means, that a symmetric distance measure is used to find the best association between all measurements and all tracks. This can be done by finding every local nearest neighbor (LNN), which can be suboptimal, or the global nearest neighbor (GNN). Where the LNN finds only the association with the minimal distance, double association may be allowed or not, the GNN finds the associations to minimize the global cost of all associations. Well known GNN methods are the Hungarian algorithm [Kuh55], or the auction algorithm [Ber88].
One big advantage of NN methods is their simplicity. In particular the implementation of LNN algorithms is quite easy and the result is comprehensible. Further, even a large amount of data is computationally feasible.
Using hard decisions in the association step works out in most cases, where the objects are well separated. In dense scenarios, the NN methods are not able to deal with ambiguities. The consequences are identifier (ID) switching or even track loss. Further, such simple tracking approaches are normally not able to estimate an existence probability of an object.

2.3.2 Probabilistic Data Association

The PDA was first presented in [BT75] with the intention to solve the problem of multiple measurements for one single object. Instead of a hard decision association, a weight is calculated for every measurement. These weights define the influence of every measurement in the estimation step of the filter cycle. Thus, using the PDA allows to circumvent the problem of hard data association by incorporating all measurements in a weighted soft decision process. One problem coming up with this kind of soft association is that even the most unlikely measurement has an influence on the new state and might lead to inaccuracies. Another drawback of this approach is the need for a computational costly calculation of the distance from every measurement to every track. In order to reduce these drawbacks to a minimum, a gating, e.g. using the Mahalanobis distance (MHD), is advisable, to neclect very unlikely pairings. The PDA is divided into three parts: the calculation of the association weights, the calculation of the estimation hypotheses, and the PDA estimation. These three parts are explained in the following subsections. A well written summary of the PDA can be found in [Gri10].

Calculating Association Weights

When calculating the PDA the association weight $\beta_{j,k}$ has to be calculated for every measurement $\boldsymbol{z}_j$, with $j = 1 \dots m_k$ at time k. This is commonly done using:

$$\beta_{j,k} = \begin{cases} \frac{b}{b+\sum_{l=1}^{m_k} e_{l,k}} & j = 0 \\ \frac{e_{j,k}}{b+\sum_{l=1}^{m_k} e_{l,k}} & j = 1, \dots, m_k \end{cases} \tag{2.31}$$

Therein, $j = 0$ represents the hypothesis, that no measurement was associated. The exponential function $e_{j,k}$ of the MHD is given with:

$$e_{j,k} = \mathrm{e}^{-\frac{1}{2} d_{j,k}^{MHD}}. \tag{2.32}$$

The MHD is given by:

$$d_{j,k}^{MHD} = \boldsymbol{\nu}_{j,k}^{\mathrm{T}} \boldsymbol{S}_{j,k}^{-1} \boldsymbol{\nu}_{j,k}, \tag{2.33}$$

where $\boldsymbol{\nu}_{j,k} = \boldsymbol{z}_{j,k} - \hat{\boldsymbol{z}}_{k|k-1}$ represents the measurement residuum and $\boldsymbol{S}_{j,k}$ is the innovation covariance matrix from Equation (2.8) for the corresponding measurement j. The weights are normalized to guarantee that they sum up to one:

$$\sum_{j=0}^{m_k} \beta_{j,k} = 1. \tag{2.34}$$

The last unknown parameter in Equation (2.31) is b, which models the influence of possible clutter measurements. Using the, not necessarily constant, detection probability p_D, the gating probability p_G, and the dimension $n_{\boldsymbol{z}}$ of the measurement space two different models are commonly used [BF88; Gri10]:

$$b = \lambda_{\copyright} \left|2\pi \boldsymbol{S}_k\right|^{\frac{1}{2}} \frac{(1 - p_D p_G)}{p_D} = \begin{cases} \left(\frac{2\pi}{\gamma}\right)^{\frac{n_z}{2}} \lambda_{\copyright} V_k c_{n_z} \frac{(1-p_D p_G)}{p_D}, \text{parametric} \\ \left(\frac{2\pi}{\gamma}\right)^{\frac{n_z}{2}} m_k c_{n_z} \frac{(1-p_D p_G)}{p_D}, \text{non-parametric} \end{cases} \tag{2.35}$$

The *parametric model* assumes the clutter measurements to be Poisson distributed. In the *non-parametric model* the clutter is expected to be equally distributed. Within Equation (2.35) $c_{n_{\boldsymbol{z}}}$ is the $n_{\boldsymbol{z}}$ dimensional unity sphere, $\lambda_{\copyright} V_k$ is the Poisson distribution parameter where V_k is the volume of the elliptical validation region of the track, and $\lambda_{\copyright}$ is the spatial density of false measurements (i.e. the average number per unit volume)[BF88], and γ is a constant parameter, which can be obtained from the quantile tables of the $n_{\boldsymbol{z}}$ dimensional chi-square distribution for constant p_G: $\gamma = \chi^2_{n_{\boldsymbol{z}}, p_G}$. Further information about the clutter models can be found in [BF88]

and [Gri10].

Calculating Estimation Hypotheses

The PDA averages over m_k hypotheses, one for every measurement $\boldsymbol{z}_{j,k}$. Every state and covariance estimation hypothesis is obtained by estimating the prediction with the standard Kalman equations from Section 2.1 with the appropriate measurement and its noise. The state hypotheses are calculated with:

$$\begin{aligned}\hat{\boldsymbol{x}}_{j,k|k} &= \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{j,k}\big[\boldsymbol{z}_{j,k} - \hat{\boldsymbol{z}}_{k|k-1}\big] \\ &= \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{j,k}\boldsymbol{\nu}_{j,k} \qquad (2.36)\\ &= \begin{cases} \hat{\boldsymbol{x}}_{k|k-1}, & j = 0 \\ \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{j,k}\boldsymbol{\nu}_{j,k}, & j = 1, \ldots, m_k \end{cases}, \qquad (2.37)\end{aligned}$$

and the equations for the covariance hypotheses are:

$$\begin{aligned}\boldsymbol{P}_{j,k|k} &= \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_{j,k}\boldsymbol{S}_{j,k}\boldsymbol{K}_{j,k}^{\mathrm{T}} \qquad (2.38)\\ &= \begin{cases} \boldsymbol{P}_{k|k-1}, & j = 0 \\ \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_{j,k}\boldsymbol{S}_{j,k}\boldsymbol{K}_{j,k}^{\mathrm{T}}, & j = 1, \ldots, m_k \end{cases}. \qquad (2.39)\end{aligned}$$

The hypothesis with $j = 0$ does not account the measurement $\boldsymbol{z}_{0,k}$, which does not exist, but represents the hypothesis, that none of the available measurements originates from the object. Thus, the hypothesis is equivalent to the prediction.

PDA Estimation

In the Estimation step of the PDA the averaging over all hypotheses is done. Since the association weights $\beta_{j,k}$ are normalized, averaging of the state is done using a weighted sum:

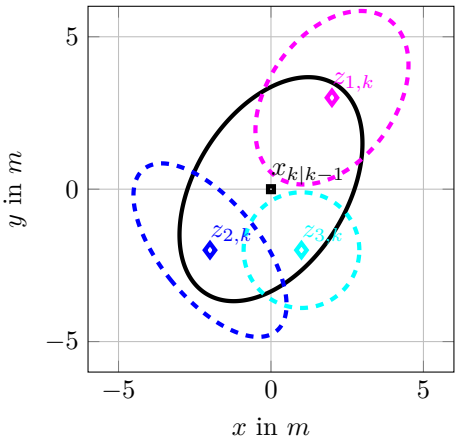
$$\boldsymbol{P}_{k|k} = \sum_{j=0}^{m_k} \underbrace{\beta_{j,k}}_{a}\Big[\underbrace{\boldsymbol{P}_{j,k|k}}_{b} + \underbrace{\big(\underbrace{\hat{\boldsymbol{x}}_{j,k|k}}_{c} - \underbrace{\hat{\boldsymbol{x}}_{k|k}}_{d}\big)\big(\hat{\boldsymbol{x}}_{j,k|k} - \hat{\boldsymbol{x}}_{k|k}\big)^{\mathrm{T}}}_{\text{correction term}}\Big] \qquad (2.40)$$

This equation can be separated into the four parts (a)-(d). Part (a) is the calculation of the association weights $\beta_{j,k}$ in Equation (2.31), (b) the estimation hypotheses of the covariance in Equation (2.39), (c) the estimation hypotheses of the state in Equation (2.37), and (d) is the estimated state. The new state (d) can be obtained

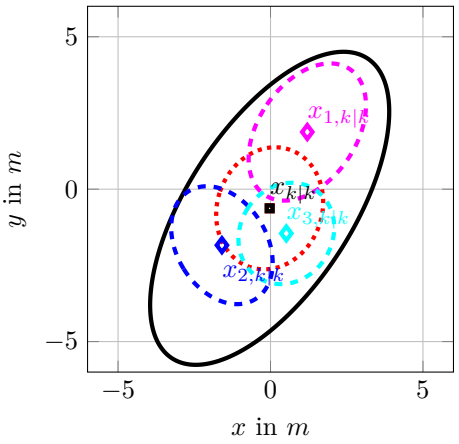
by a weighted sum over all estimation hypotheses from (c):

$$\hat{\boldsymbol{x}}_{k|k} = \sum_{j=0}^{m_k} \beta_{j,k} \hat{\boldsymbol{x}}_{j,k|k}. \tag{2.41}$$

As it can be seen in Equation (2.40), the estimation of the covariance is not only a weighted sum over all hypotheses $\boldsymbol{P}_{j,k|k}$. It is the weighted sum over the hypotheses plus a correction term. This correction term is necessary, because the weighted sum approximates multiple Gaussian distributions (the hypotheses) with a single Gaussian distribution (the new covariance and mean). This approximation leads to an underestimation of the uncertainty as depicted in Figure 2.3.



(a) Visualization of the predicted state $\boldsymbol{x}_{k|k-1}$ and three measurements $\boldsymbol{z}_{j,k}$. The black (solid) line represents the 3σ ellipse of the predicted state and the dashed lines the ellipses of the measurements.



(b) The state hypotheses for every measurement and their uncertainties (dashed). The red (dotted) ellipse represents the result of the estimation without the correction term, the black (solid) ellipse shows the correct PDA result.

Figure 2.3: Estimation step of the PDA with and without the correction term. It is obvious, that the approximation of three hypotheses with only one normal distribution is not sufficient without using the correction term.

2.3.3 Joint Integrated Probabilistic Data Association

So far, the PDA was used to track one single object observed with one or multiple measurements. In most cases the scenarios are a bit more complex, since there are often more than only one object. The PDA can be extended to the multi-object case, where so called joint events occur. These joint events are used to describe every possible combination from each track to all measurements. Therefore, Bar-Shalom et al. [BF88] introduced the joint probabilistic data association (JPDA). A further extension of the PDA is the JIPDA [ME02] which, in addition to the JPDA, integrates an estimation of an existence probability into the filter algorithm. The main task of the JIPDA is the calculation of the weights for every joint event. There are different possibilities to calculate these weights, e.g. using Murty's algorithm [Mur68]. A very intuitive way is the recursive hypothesis tree. This implementation can be found in [MSMD08]. In this graph-, or tree-, based approach the probability of every association is represented by the edge likelihood $p(e)$. An overview of all valid associations is shown in Table 2.1. An example of such a tree can be found in Figure 2.4. To determine the weights of the JIPDA there are, corresponding to

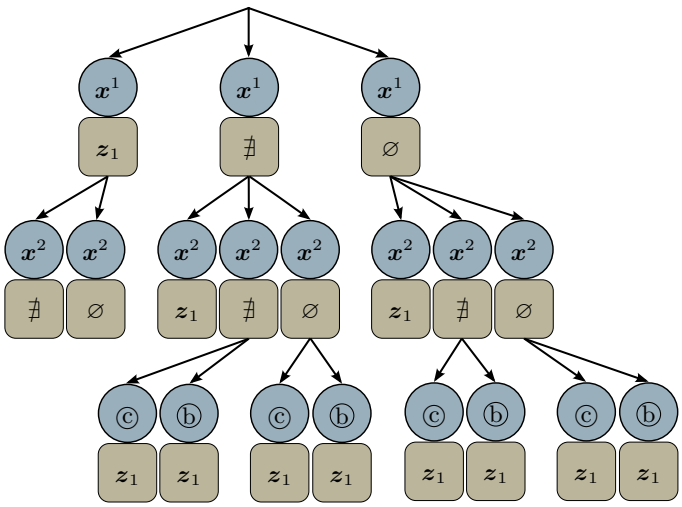


Figure 2.4: Hypothesis tree of the graph based JIPDA in case of two objects and one measurement.

Table 2.1: Valid associations of the graph based JIPDA. The birth candidates are an extension introduced in [MDM09]

$t \rightarrow j$	true positive (TP) object t generated measurement j
$\textcircled{c} \rightarrow j$	false positive (FP) measurement j is clutter
$t \rightarrow \varnothing$	false negative (FN) object t exists, but was not detected
$t \rightarrow \nexists$	true negative (TN) object t was not detected, because it does not exist
$\textcircled{b} \rightarrow j$	birth candidate (BC) measurement j is a birth candidate

Table 2.1, five different edge likelihoods:

$$p(e = (t, j)) = \Lambda_j^t \cdot p_D^t \cdot {p^t}_{k|k-1,\exists} \cdot (1 - p_{j,F}) \tag{2.42}$$

$$p(e = (\textcircled{c}, j)) = p_{j,F} \cdot \lambda_{\textcircled{c}} \tag{2.43}$$

$$p(e = (t, \varnothing)) = (1 - p_D^t) \cdot {p^t}_{k|k-1,\exists} \tag{2.44}$$

$$p(e = (t, \nexists)) = q_{\exists}^t = 1 - {p^t}_{k|k-1,\exists} \tag{2.45}$$

$$p(e = (\textcircled{b}, j)) = p_{j,\textcircled{b}}, \tag{2.46}$$

where

$$\Lambda_j^t = \frac{1}{p_G} \mathcal{N}(\boldsymbol{z}_{j,k} | \hat{\boldsymbol{z}}_{k|k-1}^t, \boldsymbol{S}_{j,k}^t) \tag{2.47}$$

is an a priori density of the measurement, p_D^t the detection probability for object $\boldsymbol{x}^t$, ${p^t}_{k,\exists}$ the predicted existence probability, $p_{j,F}$ the sensory clutter probability, $\lambda_{\textcircled{c}}$ the spatial clutter density, and $p_{j,\textcircled{b}}$ the spatial birth probability at the position of the measurement. Since most of the likelihoods are used more than once, it is advisable to store them in a look up table (LUT). For the given example, such a LUT is given in Table 2.2.

LUT	$\varnothing$	$\not\exists$	$\boldsymbol{z}_1$
ⓒ	$\times$	$\times$	$p(e = (ⓒ, 1))$
ⓑ	$\times$	$\times$	$p(e = (ⓑ, 1))$
$\boldsymbol{x}^1$	$p(e = (1, \varnothing))$	$p(e = (1, \not\exists))$	$p(e = (1, 1))$
$\boldsymbol{x}^2$	$p(e = (2, \varnothing))$	$p(e = (2, \not\exists))$	$p(e = (2, 1))$

Table 2.2: look up table of all possible edge likelihoods for the example given in Figure 2.4.

Further, Mählisch et al. proposed in [MSMD08] to go through the hypothesis tree using the recursive *Enumerate Matchings* algorithm, introduced in the same work, to accumulate all occurring likelihoods in the so called $\boldsymbol{SUM}$ table. For the given example it is given in Table 2.3. Using the $\boldsymbol{SUM}$ table, the equation of the weights

$\boldsymbol{SUM}$	$\varnothing$	$\not\exists$	$\boldsymbol{z}_1$
ⓒ	$\times$	$\times$	$\sum_{e:(ⓒ,1)\in e} p(e)$
ⓑ	$\times$	$\times$	$\sum_{e:(ⓑ,1)\in e} p(e)$
$\boldsymbol{x}^1$	$\sum_{e:(1,\varnothing)\in e} p(e)$	$\sum_{e:(1,\not\exists)\in e} p(e)$	$\sum_{e:(1,1)\in e} p(e)$
$\boldsymbol{x}^2$	$\sum_{e:(2,\varnothing)\in e} p(e)$	$\sum_{e:(2,\not\exists)\in e} p(e)$	$\sum_{e:(2,1)\in e} p(e)$

Table 2.3: $\boldsymbol{SUM}$ table with accumulated edge likelihoods for the example given in Figure 2.4.

of the JIPDA [MSMD08]

$$\beta_{j,k}^t = \frac{\sum_{e:\{t,j\}\in e} p(e)}{\sum_{e:\{t,\not\exists\}\notin e} p(e)}. \tag{2.48}$$

can be rewritten as [MSMD08]:

$$\beta_{j,k}^t = \frac{\boldsymbol{SUM}_k(t,j)}{\sum_{\iota\in\{Z\setminus\not\exists\}} \boldsymbol{SUM}_k(t,\iota)}. \tag{2.49}$$

Here Z denotes the set of all measurements including the special elements introduced in Table 2.1:

$$Z = \{\boldsymbol{z}_{1,k}, \boldsymbol{z}_{2,k}, \ldots, \boldsymbol{z}_{m_k,k}, \varnothing, \not\exists\}. \tag{2.50}$$

Having multiple objects, the weights have an additional index t in comparison to the weights of the PDA. Figure 2.5 shows a simple flow graph to visualize the data

streams (compare to the single-object single-measurement case in Figure 2.1).

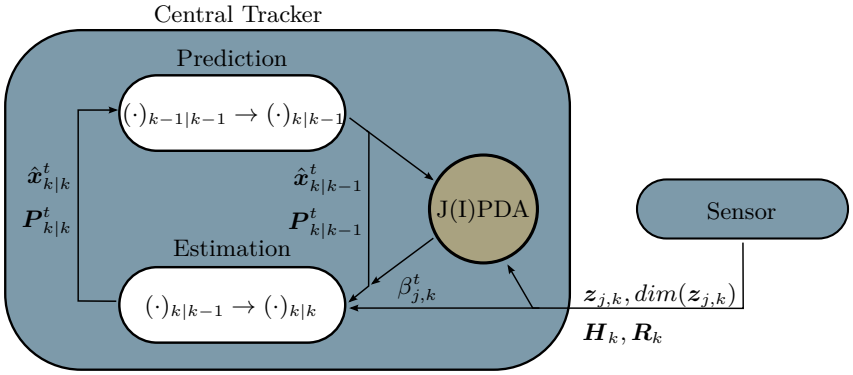


Figure 2.5: Multi-object Kalman filter in state space using J(I)PDA.

With multiple objects the PDA equations have to be adapted as well:

$$\hat{\boldsymbol{x}}^t_{j,k|k} = \hat{\boldsymbol{x}}^t_{k|k-1} + \boldsymbol{K}^t_{j,k}\boldsymbol{\nu}^t_{j,k} \tag{2.51}$$

$$\boldsymbol{P}^t_{j,k|k} = \boldsymbol{P}^t_{k|k-1} - \boldsymbol{K}^t_{j,k}\boldsymbol{S}^t_k{\boldsymbol{K}^t_{j,k}}^{\mathrm{T}} \tag{2.52}$$

$$\hat{\boldsymbol{x}}^t_{k|k} = \sum_{j=0}^{m_k} \beta^t_{j,k}\hat{\boldsymbol{x}}^t_{j,k|k} \tag{2.53}$$

$$\boldsymbol{P}^t_{k|k} = \sum_{j=0}^{m_k} \beta^t_{j,k}[\boldsymbol{P}^t_{j,k|k} + \left(\hat{\boldsymbol{x}}^t_{j,k|k} - \hat{\boldsymbol{x}}^t_{k|k}\right)\left(\hat{\boldsymbol{x}}^t_{j,k|k} - \hat{\boldsymbol{x}}^t_{k|k}\right)^{\mathrm{T}}] \tag{2.54}$$

So far, there is no prediction and estimation of existence. Thus, setting the predicted existence probability to ${p^t}_{k|k-1,\exists} = 1$, the filter is equivalent to the JPDA. To fully integrate the existence into the filter, the predicted existence probability is calculated using the first order Markov chain from Figure 2.6:

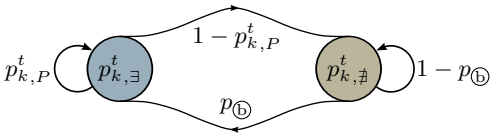


Figure 2.6: First order Markov chain of the integrated existence estimation in the prediction step.

$${p^t}_{k|k-1,\exists} = {p^t}_{k-1|k-1,\exists}p^t_{k,P} + p_{\textcircled{b}}(1 - {p^t}_{k-1|k-1,\exists}), \tag{2.55}$$

where it is a common practice to use a constant a priori birth probability $p_{ⓑ} = p_{ⓑ,prior}$ for any new object, and $p^t_{k,P}$ is the probability for object t to persist. The estimation of the existence can be done with [MSMD08]

$$p^t{}_{k|k,\exists} = \frac{\sum_{e:\{\boldsymbol{x},\nexists\}\in e} p(e)}{\sum_e p(e)} \tag{2.56}$$

using the $\boldsymbol{SUM}$ table again [MSMD08]:

$$p^t{}_{k|k,\exists} = \frac{\sum_{\iota\in\{Z\setminus\nexists\}} \boldsymbol{SUM}_k(t,\iota)}{\sum_{\iota\in\{Z\}} \boldsymbol{SUM}_k(t,\iota)}. \tag{2.57}$$

An extension of the JIPDA as presented so far is to assume that a sensor is able to provide an evidence about a measurement [Mun11]. This evidence can be expressed as a true positive (TP) probability for every measurement: $p_{j_{TP}}$. An approach to model such an evidence is to use a probability generating classification like a standard Bayes classifier (see e.g. [DHS01]). Further examples of sensor evidence models can be found in [Sto10] and [Mun11]. With this assumption Equations (2.43) and (2.46) simplify to:

$$p(e = (ⓒ, j)) = 1 - p_{j_{TP}} \tag{2.58}$$

$$p(e = (ⓑ, j)) = p_{j,k_{TP}} \cdot p_{j,ⓑ}. \tag{2.59}$$

Further, Munz proposed in [Mun11] to eliminate the spatial distribution in Equation (2.47) by replacing the a priori density Λ^t_j by the measurement likelihood $p_\Lambda(\boldsymbol{x}^t_k, \boldsymbol{z}_{j,k})$ with:

$$p_\Lambda(\boldsymbol{x}^t_k, \boldsymbol{z}_{j,k}) = \mathrm{e}^{{\boldsymbol{\nu}^t_{j,k}}^{\mathrm{T}} \; {\boldsymbol{S}^t_{j,k}}^{-1} \; \boldsymbol{\nu}^t_{j,k}}. \tag{2.60}$$

2.4 Performance Evaluation

To compare different multi-object tracking algorithms against each other, a consistent performance metric is necessary. In single-object applications, state estimation errors like the Euclidean distance or the MHD are sufficient. In multi-object scenarios, a performance metric has to incorporate the cardinality error as well. A traditional approach to that problem is the Hausdorff distance, but it is relatively insensitive to cardinality errors. This problem was addressed by the optimal mass transfer (OMAT) metric presented in [HM04]. Schuhmacher et al. have shown in [SVV08] that there are still some drawbacks using the OMAT. Some of the drawbacks are: the OMAT is undefined in case of an empty set, and it does not penalize if there are multiple

estimates for the same object. To overcome the drawbacks, Schuhmacher et al. presented the optimal subpattern assignment (OSPA) metric. A brief introduction to the OSPA metric is given in this section.

2.4.1 Optimal Subpattern Assignment

The basis of the OSPA metric is the distance d of two state vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. This distance can be calculated using a valid metric, e.g. the Euclidean distance between to states $d(\boldsymbol{x}, \boldsymbol{y})$. The OSPA metric of a single pair is this distance limited to an upper bound $\mathfrak{c}$. This bound is called the cut-off:

$$d_{\mathfrak{c}}(\boldsymbol{x}, \boldsymbol{y}) = \min(\mathfrak{c}, d(\boldsymbol{x}, \boldsymbol{y})). \tag{2.61}$$

The cut-off parameter $\mathfrak{c}$ is used to prevent that high distances between true and estimated states cause larger OSPA distances than a missed detection. In case of multiple objects, the single state vectors are replaced by two finite sets $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ and $Y = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\}$. For $m \leq n$, the OSPA distance now is obtained by:

$$d_{\mathfrak{c}}^{\mathfrak{p}}(X, Y) = \left(\frac{1}{n} \left(\min_{\pi \in \Pi_n} \sum_{i=1}^{m} d_{\mathfrak{c}}(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(\pi(i))})^{\mathfrak{p}} + \mathfrak{c}^{\mathfrak{p}}(n-m) \right) \right)^{\frac{1}{\mathfrak{p}}}. \tag{2.62}$$

Therein, Π_k is a set of possible permutations on $\{1, 2, \ldots, k\}$ for any $k \in \mathbb{N}$. In case of $m > n$, the sets X and Y have to be switched. An increasing parameter $\mathfrak{p}$ while keeping $\mathfrak{c}$ constant leads to a higher penalization of outliers. For $\mathfrak{p} = 1$, the parameter $\mathfrak{c}$ is the penalty given to any false or missing estimation. In this work, the parameter $\mathfrak{p}$ is always set to $\mathfrak{p} = 1$. For further details on the use of the parameters $\mathfrak{p}$ and $\mathfrak{c}$ refer to [SVV08].

2.4.2 Optimal Subpattern Assignment for Tracks

The OSPA metric is evaluated for every single time step. Thus, it does not incorporate any information about the continuity of a track. In many cases, it is important to have continuous track IDs. In these cases, there is the need to evaluate tracking algorithms with respect to the track continuity. This problem was addressed in [RVCV11]. Therein, the optimal subpattern assignment for tracks (OSPAT) as an extension to the OSPA metric was proposed. In the OSPAT, the state vectors are extended to labeled state vectors $\underline{\boldsymbol{x}} = (\boldsymbol{x}, l)$ and $\underline{\boldsymbol{y}} = (\boldsymbol{y}, s)$, with the labels l and s.

The distance of the new labeled state vectors are defined by the OSPAT as:

$$d\left(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}\right) = \left(d\left(\boldsymbol{x}, \boldsymbol{y}\right)^{\mathfrak{p}} + \left(\mathfrak{a}\left(1 - \delta_l\left(s\right)\right)\right)^{\mathfrak{p}}\right)^{\frac{1}{\mathfrak{p}}}. \tag{2.63}$$

Therein, the parameter $\mathfrak{a} \in [0, \mathfrak{c}]$ represents the influence of mismatching labels and $\delta_l\left(s\right)$ is a Dirac delta function. A value of $\mathfrak{a} = 0$ simplifies the OSPAT to the OSPA as explained above. Choosing $\mathfrak{a} = \mathfrak{c}$, a mismatching label causes the same distance as a missed detection.

Chapter 3

Generic Sensor Data Fusion in Information Space

The information filter as introduced in Section 2.2 is mathematically equivalent to the Kalman filter. In publications like [GDH92] and [Mut98] the main advantage of the information filter is the simple realization of a decentralized filter architecture. Every node of a decentralized network can incorporate the information transmitted in information space by only summing it up. Nevertheless, until now, it is not possible to use other methods than nearest neighbor association to fuse information from different sources in information space. As this is a major drawback, why should one use the information filter in a centralized fusion system? The answer is quite simple: it is the anonymization of the sensor. Anonymization is the crucial point to realize a completely generic sensor fusion system. The principle of this anonymization is explained in detail in this chapter.
Every type of sensor has its own measurement model. This model is necessary in the filtering process. With the measurement model, the measurement itself, its states and dimension, changes. This results in a different size of the measurement vector or different entries in it. Thus, if a sensor is exchanged the fusion system has to be adapted to the new sensor. The result of the anonymization is an unified sensor interface. Further, most sensors have an extrinsic calibration. Some also have an intrinsic one. If this calibration never changes again, the fusion system does not have to handle it. If the calibration is done very often, e.g. using an auto calibration, the calibration can be incorporated into the sensor data and therefore into the measurement model. Thus, a new calibration results again in a change of the measurement model. Another characteristic of the anonymization of the sensor is, that the fusion system does not need any detailed knowledge about the sensor or its measurement principle. Therefore, the supplier of the sensor has not to provide these details.
In the following it is explained how the information filter can be used in the JIPDA fusion system and others.

3.1 Information Filter in PDA

When calculating the PDA in information space a lot of equations are the same as in state space. Some equations, especially those depending explicitly on the measurement or the measurement model, change in information space. The standard equations of the Kalman filter in Section 2.1 are replaced by the equations of the information filter in Section 2.2. Since the prediction of the PDA does not differ from the prediction of the standard Kalman filter, the major difference to the PDA introduced in Section 2.3.2 is the estimation step.

3.1.1 Calculating Association Weights

Calculating the association weights is straight forward, since the calculation of the weights $\beta_{j,k}$ of the PDA in information space is nearly the same as in state space. There are two confinements: first, in Equation (2.35) the dimension of the measurement $n_{\boldsymbol{z}}$ is necessary. The problem in information space is, that the information gain $\boldsymbol{i}$ is transmitted instead of the measurement $\boldsymbol{z}$. Thus, the dimension of the information vector $\boldsymbol{i}$ is the same as the one of state vector $\boldsymbol{x}$. Consequently, the dimension $n_{\boldsymbol{z}}$ has to be transmitted from the sensor to the fusion system in addition. The second constraint is the calculation of the MHD in Equation (2.33). Thus, there is the need for an approach which allows the calculation of the MHD or an equivalent. Such an approach is presented below in Section 3.1.2.

3.1.2 Mahalanobis Distance

The Mahalanobis distance (MHD) is a necessary distance measure used in several equations of the PDA. Taking a closer look at the Equation (2.33), one will notice, that the MHD can not be calculated in information space. This is due to the absence of the innovation covariance, the predicted measurement, and the measurement itself. This problem was also addressed in [FD93], where Fernandez et al. proposed to use the following equivalence:

$$\boldsymbol{\eta}_{j,k}^{\mathrm{T}} \boldsymbol{\Upsilon}_{j,k}^{\dagger} \boldsymbol{\eta}_{j,k} = \boldsymbol{\nu}_{j,k}^{\mathrm{T}} \boldsymbol{S}_{j,k}^{-1} \boldsymbol{\nu}_{j,k}. \tag{3.1}$$

Therein, $\boldsymbol{\nu}_{j,k}$ is the already known measurement residuum. Further, Fernandez et al. pointed out that $\boldsymbol{\eta}_{j,k}$ is the equivalent to the measurement residuum in information

space:

$$\begin{aligned}\boldsymbol{\eta}_{j,k} &= \boldsymbol{H}_k^{\mathrm{T}} \boldsymbol{R}_k^{-1} \boldsymbol{\nu}_k \\ &= \boldsymbol{i}_{j,k} - \boldsymbol{I}_{j,k} \boldsymbol{Y}_{k|k-1}^{-1} \hat{\boldsymbol{y}}_{k|k-1}. \end{aligned} \tag{3.2}$$

Consequently, $\boldsymbol{\Upsilon}_{j,k}$ is defined as:

$$\boldsymbol{\Upsilon}_{j,k} = \boldsymbol{I}_{j,k} + \boldsymbol{I}_{j,k} \boldsymbol{Y}_{k|k-1}^{-1} \boldsymbol{I}_{j,k}. \tag{3.3}$$

Actually, what really is necessary is the inverse $\boldsymbol{\Upsilon}_{j,k}^{-1}$. Unfortunately $\boldsymbol{\Upsilon}_{j,k}$ is not always invertible, since the dimension of the observation $n_{\boldsymbol{z}}$ is in most cases smaller than the dimension of the state $n_{\boldsymbol{x}}$. Thus, the rank of $\boldsymbol{I}_{j,k}$ is not always full and therefore is not invertible in general. Consequently $\boldsymbol{\Upsilon}_{j,k}$ is not invertible in general as well, since:

$$\operatorname{rank}(\boldsymbol{\Upsilon}_{j,k}) \leq \operatorname{rank}(\boldsymbol{I}_{j,k}). \tag{3.4}$$

As solution, the Moore-Penrose pseudo-inverse [Moo20] is used, which is denoted by the † symbol. In case of an invertible matrix the inverse and the pseudo-inverse are identical.

3.1.3 Calculating Estimation Hypotheses

Calculating the estimation hypotheses needed in Equation (3.13) is very similar to the state space case. Using the estimation equations of the information filter in Equations (2.27) and (2.28), the hypotheses in Equations (2.37) and (2.39) can be rewritten. In equivalence to the hypotheses of the estimated state in Equation (2.37) the hypotheses of the estimated information vector are:

$$\hat{\boldsymbol{y}}_{j,k|k} = \hat{\boldsymbol{y}}_{k|k-1} + \boldsymbol{i}_{j,k} \tag{3.5}$$

$$= \begin{cases} \hat{\boldsymbol{y}}_{k|k-1}, & j = 0 \\ \hat{\boldsymbol{y}}_{k|k-1} + \boldsymbol{i}_{j,k}, & j = 1, \ldots, m_k \end{cases}. \tag{3.6}$$

The corresponding hypotheses of the covariance in Equation (2.39) are replaced by the hypotheses for the estimated information matrix:

$$\boldsymbol{Y}_{j,k|k} = \boldsymbol{Y}_{k|k-1} + \boldsymbol{I}_{j,k} \tag{3.7}$$

$$= \begin{cases} \boldsymbol{Y}_{k|k-1}, & j = 0 \\ \boldsymbol{Y}_{k|k-1} + \boldsymbol{I}_{j,k}, & j = 1, \ldots, m_k \end{cases}. \tag{3.8}$$

Again, the case $j = 0$ represents the hypothesis, that all measurements are clutter and none of it is assigned to the actual object.

3.1.4 PDA Estimation

The approach to calculate the estimation step of the PDA with information measurements is to slightly modify the estimation in Equation (2.40), and to replace the estimated covariance with the estimated information matrix:

$$\boldsymbol{Y}_{k|k} = \left(\sum_{j=0}^{m_k} \underbrace{\beta_{j,k}}_{a} \left[\underbrace{\boldsymbol{P}_{j,k|k}}_{b} + \big(\underbrace{\hat{\boldsymbol{x}}_{j,k|k}}_{c} - \underbrace{\hat{\boldsymbol{x}}_{k|k}}_{d} \big) \big(\hat{\boldsymbol{x}}_{j,k|k} - \hat{\boldsymbol{x}}_{k|k} \big)^{\mathrm{T}} \right] \right)^{-1}. \tag{3.9}$$

Equivalent to the equation in state space there are four parts (a)-(d). In (a), as before, the association weights are calculated. Parts (b)-(c) are rewritten in information space using Equations (2.22) to (2.24).

$$(b): \quad \boldsymbol{P}_{j,k|k} = \boldsymbol{Y}_{j,k|k}^{-1} \tag{3.10}$$

$$(c): \quad \hat{\boldsymbol{x}}_{j,k|k} = \boldsymbol{P}_{j,k|k} \hat{\boldsymbol{y}}_{j,k|k} = \boldsymbol{Y}_{j,k|k}^{-1} \hat{\boldsymbol{y}}_{j,k|k}. \tag{3.11}$$

In (d), as explained above, a weighted mean $\hat{\boldsymbol{x}}$ of all m_k estimation hypotheses is calculated. Using Equation (3.11), the Equation (2.41) can be written as:

$$(d): \quad \hat{\boldsymbol{x}}_{k|k} = \boldsymbol{Y}_{k|k}^{-1} \hat{\boldsymbol{y}}_{k|k} = \sum_{j=0}^{m_k} \beta_{j,k} \boldsymbol{Y}_{j,k|k}^{-1} \hat{\boldsymbol{y}}_{j,k|k} \tag{3.12}$$

After substituting Equations (3.10) to (3.12) into Equation (3.9), the update in information space is:

$$\boldsymbol{Y}_{k|k} = \left(\sum_{j=0}^{m_k} \beta_{j,k} \left[\boldsymbol{Y}_{j,k|k}^{-1} + \widetilde{\boldsymbol{Y}}_{j,k} \widetilde{\boldsymbol{Y}}_{j,k}^{\mathrm{T}} \right] \right)^{-1} \tag{3.13}$$

with

$$\begin{aligned} \widetilde{\boldsymbol{Y}}_{j,k} &= \hat{\boldsymbol{x}}_{j,k|k} - \hat{\boldsymbol{x}}_{k|k} \\ &= \big(\boldsymbol{Y}_{j,k|k}^{-1} \hat{\boldsymbol{y}}_{j,k|k} - \sum_{j=0}^{m_k} \beta_{j,k} \boldsymbol{Y}_{j,k|k}^{-1} \hat{\boldsymbol{y}}_{j,k|k} \big). \end{aligned} \tag{3.14}$$

3.2 Information Filter in JIPDA

As shown in Chapter 2 the main difference of JPDA and JIPDA to the single-object PDA is the calculation of the association weights. In the multi-object version, a hypothesis tree is used to determine the weight for every hypothesis. Table 2.1 shows, as already explained, the possible associations including the special symbols. The corresponding equations are given by Equations (2.42) to (2.46). Since Equations (2.44) to (2.46) do not depend on any state space parameter, these can be used in the information space form of the JIPDA without any adaption. In contrast, the TP probability $p(e = (t, j))$ in Equation (2.42) depends directly on the state space because of the measurement likelihood Λ_j^t. The FP probability in Equation (2.43) may depend on the state space because of the spatial clutter $\lambda_{\text{©}}$, depending on the implementation of the clutter model. However, most of the suggested models approximate the gating volume and therefore depend on the determinant of the innovation covariance $|\boldsymbol{S}_{j,k}^t|$ [Mäh09; MM04] which in turn depends on the state space as depicted in Equation (2.8).
The suggested solution to solve these dependencies is depicted in this section. After solving the remaining state space dependencies, the obtained algorithm is a JPDA or JIPDA which can be calculated using only information measurements. This leads to the flow graph in Figure 3.1 in comparison to the one in state space (Figure 2.5).

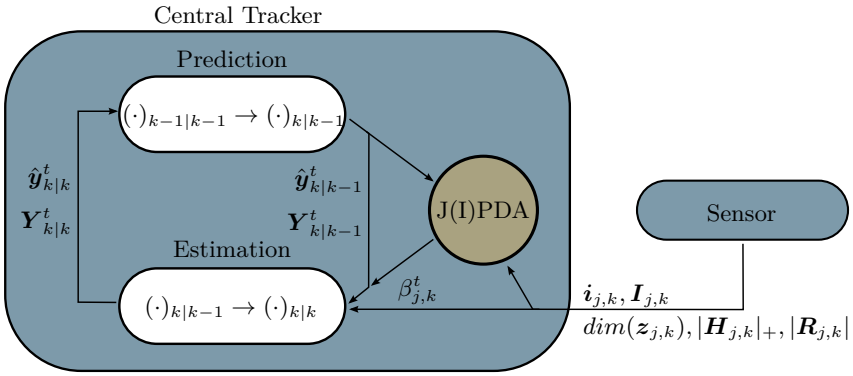


Figure 3.1: Multi-object Kalman filter in information space using J(I)PDA

3.2.1 Measurement Likelihood

As pointed out, the measurement likelihood Λ_j^t as used in Equation (2.42) is one of the remaining dependencies of the state space to be solved. As shown in Equation (2.47),

the likelihood is the result of a multivariate normal distribution $\mathcal{N}(\boldsymbol{z}_{j,k}|\hat{\boldsymbol{z}}^t_{k|k-1}, \boldsymbol{S}^t_{j,k})$ with mean $\hat{\boldsymbol{z}}^t_{k|k-1}$ and covariance $\boldsymbol{S}^t_{j,k}$. Unfortunately, none of the parameters of the normal distribution is available when using the information filter. In order eliminate these dependencies on the state it is necessary to take a look at the derivation of the likelihood, where it is obtained by the following marginalization [Mäh09]:

$$\mathcal{N}(\boldsymbol{z}_{j,k}|\hat{\boldsymbol{z}}^t_{k|k-1}, \boldsymbol{S}^t_{j,k}) = \int \mathcal{N}(\boldsymbol{z}_{j,k}|\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k, \boldsymbol{R}_{j,k})\mathcal{N}(\boldsymbol{x}^t_k|\hat{\boldsymbol{x}}^t_{k|k-1}, \boldsymbol{P}^t_{k|k-1})\mathrm{d}\boldsymbol{x}^t_k \quad (3.15)$$

In this equation, all unknown dependencies are part of the normal distribution $\mathcal{N}(\boldsymbol{z}_{j,k}|\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k, \boldsymbol{R}_{j,k})$ of the measurement. Therein, $\boldsymbol{x}^t_k$ is the true state of the observed object to be estimated. Thus, $\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k$ is the true measurement of the object, which does not exist, since it is influenced by the measurement noise $\boldsymbol{R}_{j,k}$. This results in the noise afflicted measurement $\boldsymbol{z}_{j,k}$. In the information space, there is no explicit knowledge about any of the parameters. Writing out the normal distribution of the measurement in full results in:

$$\begin{aligned}&\mathcal{N}(\boldsymbol{z}_{j,k}|\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k, \boldsymbol{R}_{j,k})\\ &= \frac{1}{\sqrt{(2\pi)^{n_{\boldsymbol{z}}}|\boldsymbol{R}_{j,k}|}}\mathrm{e}^{-\frac{1}{2}\left(\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k-\boldsymbol{z}_{j,k}\right)^{\mathrm{T}}\boldsymbol{R}^{-1}_{j,k}\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k-\boldsymbol{z}_{j,k}\right)\right)}\end{aligned} \quad (3.16)$$

Therein, several parameters are unknown in information space. To the knowledge of the author, there is no possibility to determine the dimension of the measurement $n_{\boldsymbol{z}}$ and the determinant of the measurement uncertainty $|\boldsymbol{R}_{j,k}|$ using the information contribution $\boldsymbol{i}_{j,k}$ and $\boldsymbol{I}_{j,k}$. These two parameters have to be provided by the sensor. Since these are only two single values, the overhead when transmitting them is acceptable. Further, it is not possible to use these information to recover the measurement principle of the sensor. In contrast, the exponent of the right hand side in Equation (3.16) contains explicit information about the sensor. Taking a closer look at the exponent allows some useful conversions:

$$\begin{aligned}&\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k - \boldsymbol{z}_{j,k}\right)^{\mathrm{T}}\boldsymbol{R}^{-1}_{j,k}\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k - \boldsymbol{z}_{j,k}\right)\\ &= \left(\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k\right)^{\mathrm{T}} - \boldsymbol{z}^{\mathrm{T}}_{j,k}\right)\boldsymbol{R}^{-1}_{j,k}\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k - \boldsymbol{z}_{j,k}\right)\\ &= \left({\boldsymbol{x}^t_k}^{\mathrm{T}}\,\boldsymbol{H}^{\mathrm{T}}_{j,k} - \boldsymbol{z}^{\mathrm{T}}_{j,k}\right)\boldsymbol{R}^{-1}_{j,k}\left(\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k - \boldsymbol{z}_{j,k}\right)\\ &= {\boldsymbol{x}^t_k}^{\mathrm{T}}\,\boldsymbol{H}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k - {\boldsymbol{x}^t_k}^{\mathrm{T}}\,\boldsymbol{H}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{z}_{j,k} - \boldsymbol{z}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k + \boldsymbol{z}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{z}_{j,k}\\ &= {\boldsymbol{x}^t_k}^{\mathrm{T}}\,\boldsymbol{I}_{j,k}\boldsymbol{x}^t_k - {\boldsymbol{x}^t_k}^{\mathrm{T}}\,\boldsymbol{i}_{j,k} - \underbrace{\boldsymbol{z}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k}_{a} + \underbrace{\boldsymbol{z}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{z}_{j,k}}_{b}\end{aligned} \quad (3.17)$$

For further conversions of the parts (a) and (b) in Equation (3.17) some assumptions are necessary. In the following it is assumed, that $\boldsymbol{R}_{j,k}$ and $\boldsymbol{I}_{j,k}$ are symmetric. This

does not cause further restrictions. In addition it is necessary to regard $\boldsymbol{H}_{j,k}$,$\boldsymbol{R}_{j,k}$, and $\boldsymbol{I}_{j,k}$ as invertible. For $\boldsymbol{R}_{j,k}$ this has no consequences. In contrast, this assumption is not always true for $\boldsymbol{H}_{j,k}$ and $\boldsymbol{I}_{j,k}$. This problem is referred to later in this section. Now, further conversions of part (a) in Equation (3.17) can be done:

$$\begin{aligned}
&\boldsymbol{z}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \boldsymbol{H}_{j,k} \boldsymbol{x}_k^t \\
&= \boldsymbol{z}_{j,k}^{\mathrm{T}} \left(\boldsymbol{H}_{j,k}^{-1} \boldsymbol{R}_{j,k} \right)^{-1} \boldsymbol{x}_k^t \\
&= \boldsymbol{z}_{j,k}^{\mathrm{T}} \left(\boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-\mathrm{T}} \right)^{\mathrm{T}} \boldsymbol{x}_k^t \\
&= \left(\boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \boldsymbol{z}_{j,k} \right)^{\mathrm{T}} \boldsymbol{x}_k^t \\
&= \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{x}_k^t .
\end{aligned} \tag{3.18}$$

Same assumptions used in part (b) in Equation (3.17) results in:

$$\boldsymbol{z}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \boldsymbol{z}_{j,k} = \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \boldsymbol{i}_{j,k}. \tag{3.19}$$

Using Equations (3.18) and (3.19), Equation (3.17) simplifies to

$$\begin{aligned}
&\left(\boldsymbol{H}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{z}_{j,k} \right)^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \left(\boldsymbol{H}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{z}_{j,k} \right) \\
&= {\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{I} \boldsymbol{x}_k^t - {\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{i}_{j,k} - \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{x}_k^t + \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \boldsymbol{i}_{j,k} \\
&= {\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - {\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{i}_{j,k} - \left({\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{i}_{j,k} \right)^{\mathrm{T}} + \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \boldsymbol{i}_{j,k} \\
&= {\boldsymbol{x}_k^t}^{\mathrm{T}} \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) - \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{x}_k^t + \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \boldsymbol{i}_{j,k} \\
&= {\boldsymbol{x}_k^t}^{\mathrm{T}} \left(\boldsymbol{i}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) + \boldsymbol{i}_{j,k}^{\mathrm{T}} \left(-\boldsymbol{x}_k^t + \boldsymbol{I}_{j,k}^{-1} \boldsymbol{i}_{j,k} \right) \\
&= {\boldsymbol{x}_k^t}^{\mathrm{T}} \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) - \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) \\
&= \left({\boldsymbol{x}_k^t}^{\mathrm{T}} - \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \right) \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) \\
&= \left({\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{\mathrm{T}} - \boldsymbol{i}_{j,k}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \boldsymbol{I}_{j,k}^{\mathrm{T}} \right) \boldsymbol{I}_{j,k}^{-\mathrm{T}} \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) \\
&= \left({\boldsymbol{x}_k^t}^{\mathrm{T}} \boldsymbol{I}_{j,k}^{\mathrm{T}} - \boldsymbol{i}_{j,k}^{\mathrm{T}} \right)^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-\mathrm{T}} \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right) \\
&= \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right)^{\mathrm{T}} \boldsymbol{I}_{j,k}^{-1} \left(\boldsymbol{I}_{j,k} \boldsymbol{x}_k^t - \boldsymbol{i}_{j,k} \right)
\end{aligned} \tag{3.20}$$

With the equality in Equation (3.20) the normal distribution in Equation (3.16) can be expressed as:

$$\mathcal{N}(\boldsymbol{z}_{j,k} | \boldsymbol{H}_{j,k} \boldsymbol{x}_k^t, \boldsymbol{R}_{j,k}) = c_{corr} \cdot \mathcal{N}(\boldsymbol{i}_{j,k} | \boldsymbol{I}_{j,k} \boldsymbol{x}_k^t, \boldsymbol{I}_{j,k}) \tag{3.21}$$

where c_{corr} is a correction term:

$$c_{corr} = \frac{\sqrt{(2\pi)^{n_{\boldsymbol{i}}} \left|\boldsymbol{I}_{j,k}\right|_+}}{\sqrt{(2\pi)^{n_{\boldsymbol{z}}} \left|\boldsymbol{R}_{j,k}\right|}} = \sqrt{(2\pi)^{n_{\boldsymbol{i}}-n_{\boldsymbol{z}}} \frac{\left|\boldsymbol{I}_{j,k}\right|_+}{\left|\boldsymbol{R}_{j,k}\right|}}. \tag{3.22}$$

Since the state $\boldsymbol{x}$ has the same dimension as $\boldsymbol{y}$ and $\boldsymbol{i}$, it is $n_{\boldsymbol{i}} = n_{\boldsymbol{x}}$. The incorrect assumption that $\boldsymbol{H}_{j,k}$ and $\boldsymbol{I}_{j,k}$ are invertible leads to problems when calculating Equation (3.22). Therefore it is suggested to use the Moore-Penrose pseudo-inverse [Moo20] $(\cdot)^\dagger$. Further, since $\boldsymbol{I}$ is not necessarily a square matrix, the pseudo-determinant $|\cdot|_+$ presented in [Min98] should be used. Having a representation of Equation (3.16) using information measurements, it is possible to marginalize Equation (3.15) using Equations (3.21) to (3.22) to obtain the measurement likelihood in information space:

$$\begin{aligned}
&\mathcal{N}(\boldsymbol{z}_{j,k}|\hat{\boldsymbol{z}}^t_{k|k-1}, \boldsymbol{S}^t_{j,k}) \\
&= \int \mathcal{N}(\boldsymbol{z}_{j,k}|\boldsymbol{H}_{j,k}\boldsymbol{x}^t_k, \boldsymbol{R}_{j,k}) \mathcal{N}(\boldsymbol{x}^t_k|\hat{\boldsymbol{x}}^t_{k|k-1}, \boldsymbol{P}^t_{k|k-1}) \mathrm{d}\boldsymbol{x}^t_k \\
&= c_{corr} \cdot \int \mathcal{N}(\boldsymbol{i}_{j,k}|\boldsymbol{I}_{j,k}\boldsymbol{x}^t_{k|k-1}, \boldsymbol{I}_{j,k}) \mathcal{N}(\boldsymbol{x}^t_k|\hat{\boldsymbol{x}}^t_{k|k-1}, \boldsymbol{P}^t_{k|k-1}) \mathrm{d}\boldsymbol{x}^t_k \\
&= c_{corr} \cdot \mathcal{N}(\boldsymbol{i}_{j,k}|\boldsymbol{I}_{j,k}\hat{\boldsymbol{x}}^t_{k|k-1}, \boldsymbol{I}_{j,k}\boldsymbol{P}^t_{k|k-1}\boldsymbol{I}^{\mathrm{T}}_{j,k} + \boldsymbol{I}_{j,k}) \\
&= c_{corr} \cdot \mathcal{N}(\boldsymbol{i}_{j,k}|\boldsymbol{I}_{j,k}{\boldsymbol{Y}^t_{k|k-1}}^{-1}\hat{\boldsymbol{y}}^t_{k|k-1}, \boldsymbol{I}_{j,k}{\boldsymbol{Y}^t_{k|k-1}}^{-1}\boldsymbol{I}^{\mathrm{T}}_{j,k} + \boldsymbol{I}_{j,k}).
\end{aligned} \tag{3.23}$$

Again, the pseudo-inverse $(\cdot)^\dagger$ and the pseudo-determinant $|\cdot|_+$ should be used to calculate the distributions for the above mentioned reasons.

3.2.2 Gating Volume

As explained above, the determinant of the innovation covariance $|\boldsymbol{S}^t_{j,k}|$ is used in JIPDA to approximate the gating volume. In turn, the gating volume is often used in clutter models. However, when tracking in the information space the innovation covariance is unknown. Using the same assumption as in Section 3.2.1, that $\boldsymbol{H}_{j,k}$ and $\boldsymbol{I}_{j,k}$ are invertible, it is possible to obtain an approximation. Thus, the following conversion is feasible:

$$\begin{aligned}
\boldsymbol{\Upsilon}^t_{j,k} &= \boldsymbol{I}_{j,k} + \boldsymbol{I}_{j,k}{\boldsymbol{Y}^t_{k|k-1}}^{-1}\boldsymbol{I}_{j,k} \\
&= \boldsymbol{I}_{j,k} + \boldsymbol{I}_{j,k}{\boldsymbol{P}^t_{k|k-1}}^{-1}\boldsymbol{I}_{j,k} \\
&= \boldsymbol{H}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{H}_{j,k} + \boldsymbol{H}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{H}_{j,k}\boldsymbol{P}_{k|k-1}\boldsymbol{H}^{\mathrm{T}}_{j,k}\boldsymbol{R}^{-1}_{j,k}\boldsymbol{H}_{j,k}
\end{aligned}$$

$$
\begin{aligned}
&= \boldsymbol{H}_{j,k}^{\mathrm{T}} \left(\boldsymbol{R}_{j,k}^{-1} \boldsymbol{H}_{j,k} + \boldsymbol{R}_{j,k}^{-1} \boldsymbol{H}_{j,k} \boldsymbol{P}_{k|k-1} \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \; \boldsymbol{H}_{j,k} \right) \\
&= \boldsymbol{H}_{j,k}^{\mathrm{T}} \left(\boldsymbol{R}_{j,k}^{-1} + \boldsymbol{R}_{j,k}^{-1} \boldsymbol{H}_{j,k} \boldsymbol{P}_{k|k-1} \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \right) \boldsymbol{H}_{j,k} \\
&= \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \left(1 + \boldsymbol{H}_{j,k} \boldsymbol{P}_{k|k-1} \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \right) \boldsymbol{H}_{j,k} \\
&= \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \left(\boldsymbol{R}_{j,k} + \boldsymbol{H}_{j,k} \boldsymbol{P}_{k|k-1} \boldsymbol{H}_{j,k}^{\mathrm{T}} \right) \boldsymbol{R}_{j,k}^{-1} \; \boldsymbol{H}_{j,k} \\
&= \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \boldsymbol{H}_{j,k} \boldsymbol{H}_{j,k}^{-1} \boldsymbol{S}_{j,k}^{t} \boldsymbol{H}_{j,k}^{-\mathrm{T}} \boldsymbol{H}_{j,k}^{\mathrm{T}} \boldsymbol{R}_{j,k}^{-1} \boldsymbol{H}_{j,k} \\
&= \boldsymbol{I}_{j,k} \boldsymbol{H}_{j,k}^{-1} \boldsymbol{S}_{j,k}^{t} \boldsymbol{H}_{j,k}^{-\mathrm{T}} \boldsymbol{I}_{j,k}^{\mathrm{T}}. \qquad (3.24)
\end{aligned}
$$

Since it is only the determinant instead of the complete matrix, the searched expression for $|\boldsymbol{S}_{j,k}^{t}|$ is:

$$|\boldsymbol{\Upsilon}_{j,k}^{t}| = |\boldsymbol{I}_{j,k}||\boldsymbol{H}_{j,k}^{-1}||\boldsymbol{S}_{j,k}^{t}||\boldsymbol{H}_{j,k}^{-\mathrm{T}}||\boldsymbol{I}_{j,k}^{\mathrm{T}}| \qquad (3.25)$$

$$|\boldsymbol{S}_{j,k}^{t}| = \frac{|\boldsymbol{\Upsilon}_{j,k}^{t}||\boldsymbol{H}_{j,k}|_{+}^{2}}{|\boldsymbol{I}_{j,k}|_{+}^{2}} \qquad (3.26)$$

As one can see in Equation (3.26), it is not possible to calculate $|\boldsymbol{S}_{j,k}^{t}|$ using available items solely, since $|\boldsymbol{H}_{j,k}|_{+}$ is unknown. Therefore, this information has to be provided by the sensor. Since a determinant equals a single value, the overhead is again acceptable and it is not possible to obtain the sensor model by knowing the determinant of the measurement matrix only.

3.2.3 Using the Sensor Evidence

When using a sensor with an integrated evidence model as proposed in [Mun11], the clutter probability in Equation (2.58) is slightly different to the original one in Equation (2.43). Assuming the sensor to be able to provide a TP likelihood has the effect, that the approximation of the gating volume in Section 3.2.2 is unnecessary. Hence, it is not necessary to transmit the determinant of the measurement model $|\boldsymbol{H}_{j,k}|_{+}$ from the sensor to the central tracker. Instead, the TP likelihood $p_{j,k_{TP}}$ has to be communicated.

3.2.4 Data Transmission

A very interesting question after discussing the data to be provided by the sensors is, how much data really has to be transmitted. As depicted in the flow charts of the

JIPDA in the state space, see Figure 2.5, and the information space, see Figure 3.1, the data to transmit is: $\boldsymbol{H}_k, \boldsymbol{R}_k, \boldsymbol{z}_{j,k}$ in state space, assuming that $dim(\boldsymbol{z}_{j,k})$ is known because of the size of the received data. In information space, the transmitted data is $\boldsymbol{i}_{j,k}, \boldsymbol{I}_{j,k}, dim(\boldsymbol{z}_{j,k}), |\boldsymbol{H}_{j,k}|_+, |\boldsymbol{R}_{j,k}|$. The comparison for different sets of state and measurement dimensions are given in Table 3.1. It is shown, that the advantage

		n_z						
		state space						information space
		1	2	3	4	5	6	1-6
n_x	3	5	12	21	32	45	60	15
	4	6	14	24	36	50	66	23
	5	7	16	27	40	55	72	33
	6	8	18	30	44	60	78	45
	7	9	20	33	48	65	84	59

Table 3.1: Amount of transmitted values for every measurement depending on the dimension of the state vector $\boldsymbol{x}$ and the measurement vector $\boldsymbol{z}$ (worst case).

of the information space is, that the amount of transmitted values depends only on the dimension of the state. This is a disadvantage at the same time, since in setups, where the measurement dimension is low, the amount of values is higher than in the state space. In such cases the bigger amount of data is the tradeoff to gain a generic sensor linkage.

3.2.5 Linearization of Non-Linear Measurement Models

In case of non-linear measurement models, one certain problem arises: the linearization. In the state space, the predicted state is transformed into the measurement space using the measurement matrix $\boldsymbol{H}_k$. A non-linear model causes a measurement function $\boldsymbol{h}_k(\hat{\boldsymbol{x}})$. Using an extended or unscented Kalman filter, the measurement function is used to transform the state into measurement space, but to calculate the innovation covariance in Equation (2.8) a matrix is necessary. In the EKF for example, this problem is solved using the linearization of the measurement function at the predicted state:

$$\boldsymbol{h}_{\boldsymbol{x},k+1} = \left[\nabla_{\boldsymbol{x}} \boldsymbol{h}_{k+1}(\boldsymbol{x})^{\mathrm{T}}\right]^{\mathrm{T}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k+1|k}} := \frac{\partial \boldsymbol{h}_{k+1}(\boldsymbol{x})}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k+1|k}} \tag{3.27}$$

Using the information filter, where the fusion system does not know the measurement model, the linearization has to be done by the sensor.

Linearization Using the Predicted State As it can be seen in Equation (3.27), the linearization depends on the predicted state. Therefore, a backward channel from the fusion system to the sensor is necessary. The information sensor then has to calculate the information gain and it's corresponding information matrix with:

$$\boldsymbol{\nu}_k = \boldsymbol{z}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_{k+1|k}) \tag{3.28}$$

$$\boldsymbol{I}_k = \boldsymbol{h}_{\boldsymbol{x},k+1}^{\mathrm{T}} \boldsymbol{R}_k^{-1} \boldsymbol{h}_{\boldsymbol{x},k+1} \tag{3.29}$$

$$\boldsymbol{i}_k = \boldsymbol{h}_{\boldsymbol{x},k+1}^{\mathrm{T}} \boldsymbol{R}_k^{-1} [\boldsymbol{\nu}_k + \boldsymbol{h}_{\boldsymbol{x},k+1} \hat{\boldsymbol{x}}_{k+1|k}]. \tag{3.30}$$

In Equations (3.28) to (3.30) a predicted state is necessary to calculate the measurement. In multi-object and multi-measurement scenarios a certain problem arises: the linearization has to be done for every measurement with the corresponding predicted state. In case of a feasible backward channel some kind of measurement to track association has to be calculated. Therefore, every measurement is linearized using every predicted track. Afterwards, a nearest neighbor association using the MHD is done. The transmitted measurements are those with the smallest MHD to the predicted tracks. Here, a gating should be used in case of measurements which are caused from an object not yet tracked. This causes the next barrier: if there is no object tracked yet, a predicted state is not present. Without a predicted state, the linearization is not possible. This corresponds to the case, that sometimes a backward channel is either not feasible or not intended by the system design. It is important to mention, that the measurement function is not separable from the state [MA97] [VW05] [Lee08]. Bar-Shalom et al. also pointed out in [BL95], that the calculation of the covariances is not decoupled from the estimated states. This approach is used in the EKF of the evaluation in Section 3.4.1. Another drawback of this approach is the large amount of data to transmit to the sensor e.g. in the case of an UKF. Using this filter, the innovation covariance is calculated using a weighted sum of sigma points. To do so, all sigma points have to be transmitted to the sensor.

Linearization Using the Invertable Measurement Function In case of an invertible measurement function, one solution is, to linearize the measurement function at the position of the measurement converted into state space. This is comparable to the converted measurement Kalman filter (CMKF) presented in [BL95]. Therein, the example of a sensor which measures the radial distance and the angle is given. The measurement of this sensor can easily be transformed to Cartesian coordinates and the linearization, or the use of a linear measurement model, is possible. This solution is also evaluated in Section 3.4.1.

Linearization Using the Measurement Function with Additional Assumptions If the measurement function is not invertible, it might be possible to invert it partially to obtain only the necessary states for linearization. Measurement functions containing projections, like the pinhole camera model [HZ04], depend from several state variables. The measurement function then is not even partially invertible. In this case, additional assumptions might be the solution. Using the pinhole camera model, one approach is to assume a certain width of the perceived object to allow a transformation from the measurement space into state space or to assume the world to be absolutely flat.

3.2.6 The Continuity Problem in Angle Measurements

In case of a sensor which supplies angle measurements, e.g. because of a radial measurement principle or the yaw angle of a perceived object, a problem comes up: the discontinuity of the angle. Since the angle is always limited to a certain interval, e.g. $]-\pi, \pi]$ or $[0, 2\pi[$, the problem arises always at the borders of the interval. Figure 3.2 shows an example with the interval $[0, 2\pi[= [0°, 360°[$. In the prediction step in Equation (2.5) or the estimation step in Equation (2.10) of the Kalman filter it is not guaranteed that the intervals are not exceeded. Figure 3.2 depicts the problem of the estimation step in Equation (2.10). There, the difference of the predicted measurement $\hat{\boldsymbol{z}}_{k|k-1}$ and the current measurement $\boldsymbol{z}_{1,k}$ has to be calculated. If the sensor provides an angle, this difference inherits a difference of angles. In the example in Figure 3.2 these angles are: $\psi_{\hat{\boldsymbol{z}}_{k|k-1}} = 10°$ and $\psi_{\boldsymbol{z}_{1,k}} = 350°$. The difference then is: $\psi_{\boldsymbol{z}_{1,k}} - \psi_{\hat{\boldsymbol{z}}_{k|k-1}} = 340°$. Obviously, the difference should be 20°. In the state space such problems are solved by normalizing all angles and angle differences. In the information space a normalization is not feasible, since it is not possible to obtain the single state variables from the information gain $\boldsymbol{i}_{j,k}$. One

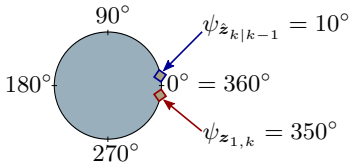


Figure 3.2: Visualization of the angle continuity problem in the interval $[0, 2\pi[$.

way to solve the problem of discontinuity in the information space is to convert the predicted information vector $\hat{\boldsymbol{y}}_{k|k-1}$ into the state space $\hat{\boldsymbol{x}}_{k|k-1}$. This usually means no computational overhead, since the complete prediction step is done in the state space even when using the information filter. Afterwards multiple predicted states

are hypothesized:

$$\hat{\boldsymbol{x}}_{k|k}^{0}; \quad \hat{\boldsymbol{x}}_{k|k}^{+2\pi}; \quad \hat{\boldsymbol{x}}_{k|k}^{-2\pi}$$

where $\pm 2\pi$ is added to the angle of the state. Now, the MHD is calculated for the three hypotheses and every information measurement $\boldsymbol{i}_{j,k}$. Here, the MHD presented in Section 3.1.2 is used. The angle hypothesis with the smallest MHD is used in the estimation step. After estimating the information vector, the added $\pm 2\pi$ have to be subtracted again.

3.3 Information Filter in other Fusion Systems

The use of the information filter approach in JIPDA is a special case because of the averaging over multiple estimation hypotheses. In other fusion systems, the information filter approach in multi-object scenarios is straight forward to the single-object case, since a hard decision association is used in most cases. The only barrier is the measurement likelihood

$$p(\hat{\boldsymbol{z}}_{k|k-1}|\boldsymbol{z}_k, \boldsymbol{R}_k) \tag{3.31}$$

which has to be evaluated in nearly every fusion system. Further, most tracking approaches assume this likelihood to be a multi-dimensional Gaussian:

$$p(\hat{\boldsymbol{z}}_{k|k-1}|\boldsymbol{z}_k, \boldsymbol{R}_k) \sim \mathcal{N}(\hat{\boldsymbol{z}}_{k-1}|\boldsymbol{z}_k, \boldsymbol{R}_k) \tag{3.32}$$

$$= \frac{1}{\sqrt{(2\pi)^{n_{\boldsymbol{z}}}|\boldsymbol{R}_k|}} \mathrm{e}^{-\frac{1}{2}\left((\boldsymbol{H}_k\boldsymbol{x}_{k|k-1}-\boldsymbol{z}_k)^{\mathrm{T}}\boldsymbol{R}_k^{-1}(\boldsymbol{H}_k\boldsymbol{x}_{k|k-1}-\boldsymbol{z}_k)\right)}. \tag{3.33}$$

Equivalent to Equation (3.21), the exponent of Equation (3.33) can be transformed as follows:

$$(\boldsymbol{H}\boldsymbol{x}-\boldsymbol{z})^{\mathrm{T}}\,\boldsymbol{R}^{-1}\,(\boldsymbol{H}\boldsymbol{x}-\boldsymbol{z}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{x} - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{i} - \boldsymbol{i}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{i}^{\mathrm{T}}\boldsymbol{I}^{\dagger}\boldsymbol{i}. \tag{3.34}$$

With that, it is possible to calculate the measurement likelihood except the leading normalization constant:

$$\frac{1}{\sqrt{(2\pi)^{n_{\boldsymbol{z}}}|\boldsymbol{R}_k|}} \tag{3.35}$$

In case of sequential Monte Carlo (SMC) methods, this constant is the same for every particle. Since the particles are normalized, this constant can be ignored. In all other cases, the normalization can be calculated with knowledge about the determinant

$|\boldsymbol{R}_k|$. This single value has to be provided by the sensor system. Further, since the inverse $\boldsymbol{I}_k^{-1}$ is not always possible, the pseudo-inverse $\boldsymbol{I}_k^{\dagger}$ has to be used.

3.4 Evaluation

To evaluate if the presented information filter approach is equivalent to the Kalman filter, three scenarios are used. Therein are two simulations: one with a linear measurement model and one with a non-linear one, as well as one real-data scenario where multiple pedestrians are tracked indoors. To guarantee that differences between state space and information space are only due to numerical reasons, new born tracks are initialized in one single step. Further, both methods are parameterized identically with constant probabilities for p_t^D, p_j^F, and $p_{\textcircled{b},prior}$. As described above, the only quantities transmitted from the information sensor to the fusion module are: $\boldsymbol{i}_j$, $\boldsymbol{I}_j$, $|\boldsymbol{R}_j|$, $|\boldsymbol{H}_j|_+$, and $n_{\boldsymbol{z}_j}$. To evaluate the difference between the information filter and the Kalman filter, the OSPA distances $\mathrm{d}_{\mathrm{O}}^{\mathrm{KF}}$ of the Kalman filter and $\mathrm{d}_{\mathrm{O}}^{\mathrm{IF}}$ of the information filter introduced in Section 2.4 are used. To penalize ID switching errors the OSPAT is used. The evaluation of every scenario consists of two figures: in (a) the OSPAT distance itself is plotted. Since there is no obvious difference at the chosen scale, only the information space result is shown. In (b) the mathematical difference $\mathrm{d}_{\mathrm{O}}^{\mathrm{KF}} - \mathrm{d}_{\mathrm{O}}^{\mathrm{IF}}$ of the OSPAT distances is depicted. The JIPDA used in this evaluation is not optimized in any way and therefore it is not the algorithm itself which is evaluated here. The difference between information and state space is evaluated to proof the equivalence of both approaches.

3.4.1 Simulation Results

In this section the two simulation scenarios are evaluated using the OSPA distances in the state and information space. In the simulation, four objects are moving along the trajectories depicted in Figure 3.3. The trajectories cross each other in one point at the same time and only differ because of the added measurement noise. Three of them are moving side by side over a short time. This scenario is chosen to show the difference in a high density scenario where all measurements are very close to each other. The used process model is a constant velocity and constant turn (CVCY) with the state vector $\boldsymbol{x} = [x\ y\ v\ \psi\ \omega]^{\mathrm{T}}$. Therein, x and y denote the Cartesian coordinates, v the velocity, ψ the yaw angle, and ω the yaw rate. As measurement model, a linear model with $\boldsymbol{z} = [x\ y\ \psi]^{\mathrm{T}}$ and a non-linear model with $\boldsymbol{z} = [r\ \varphi\ \psi]^{\mathrm{T}}$ are investigated. Herein, r is the radial distance and φ the the measurement angle. The chosen simulation parameters are given in Table 3.2. Therein, the σ-values are

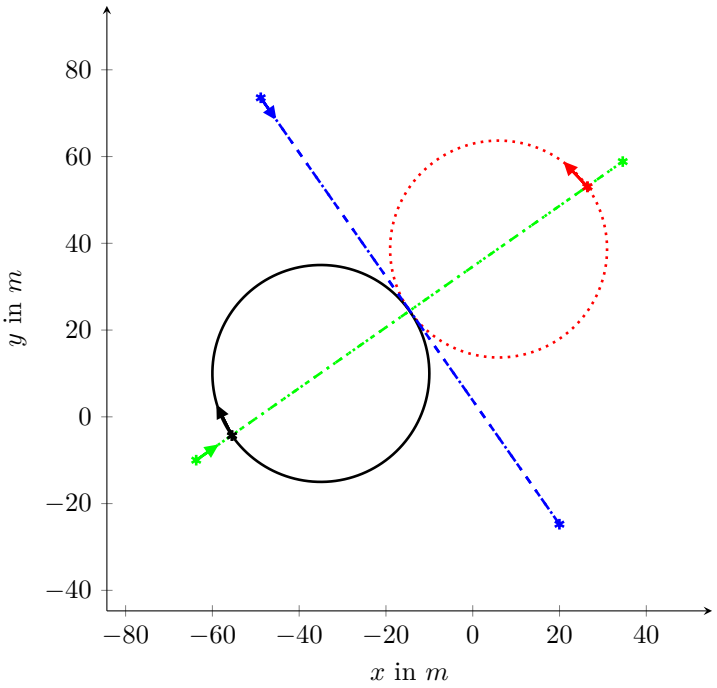


Figure 3.3: Reference trajectories of the perceived objects in the simulation scenarios. The starting point and the endpoint are marked with asterisks. The small arrows denote the starting point and yaw angle. All four objects are passing the the coordinate $[-15\ 24]^{\mathrm{T}}$ at the same time.

the noise parameters of the model and the sensor. Additionally, the simulated noise was set to the identical values than the measurement noise of the sensor. In case of the non-linear model a backward channel is used to allow the linearization in the sensor.
Figure 3.4 shows the evaluation results in case of the linear measurement model. It is illustrated that the results of the JIPDA in the state space and the here proposed version in information space are nearly identical. The maximum of the differences of both versions is in the range of 10^{-6} and gets even lower after the initialization step. Because of the mathematical equivalence of both versions in case of linear measurement models, this difference is assumed to inherit from computational inaccuracies in calculation. The evaluation of the non-linear measurement model in Figure 3.5b shows a much higher difference between state and information space. It is still quite small, and there seems to be no perceptible difference, but there are divergences which are not negligible. The most probable assumption about

Part	Parameters
Process	$\sigma_a = 6.0\frac{m}{s^2}$; $\sigma_\omega = 0.2618\frac{1}{s^2}$
Tracker	$p_D = 0.8$; $p_G = 0.95$; $p_P = 0.99$; $p_{ⓑ,prior} = 0.2$ track is born if: $p_ⓑ > 0.5$ track is killed if: $p_{k\|k,\exists} < 0.2$
Simulation	timestep $T = 0.1s$ linear: $\sigma_x = 0.5m$; $\sigma_y = 0.5m$; $\sigma_\psi = 0.06\frac{1}{s}$ non-linear: $\sigma_r = 0.3m$; $\sigma_\varphi = 0.0088$; $\sigma_\psi = 0.06\frac{1}{s}$
Sensor	linear: $\sigma_x = 0.5m$; $\sigma_y = 0.5m$; $\sigma_\psi = 0.06\frac{1}{s}$ non-linear: $\sigma_r = 0.3m$; $\sigma_\varphi = 0.0088$; $\sigma_\psi = 0.06\frac{1}{s}$
OSPAT	$\mathfrak{a} = 10$; $\mathfrak{c} = 10$; $\mathfrak{p} = 1$

Table 3.2: Simulation and tracking parameters for the evaluation of the linear and non-linear measurement model.

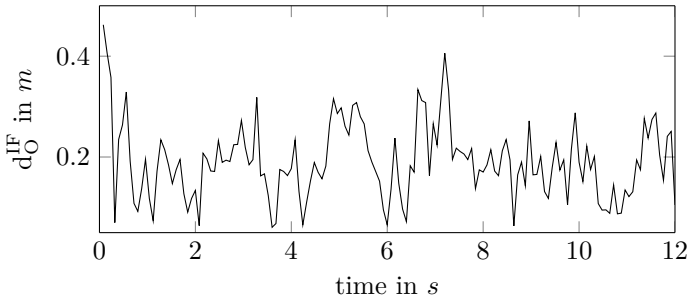
the reason of the difference is the use of the pseudo-determinant and the pseudo-inverse since the remaining parts of the algorithm are mathematically equivalent. Therefore, the error of the proposed method depends on the strength of the non-linearity. It is hardly possible to predict the expected error since the pseudo-inverse is exactly the same as the inverse in the linear case and in the non-linear case there is no real inverse to compare with. Further, the difference between the OSPAT distances of the linearization at the predicted track (TL) and the linearization at the transformed measurement (ML) is evaluated in Figure 3.5c. The distances itself are depicted in Figure 3.5a. The evaluation of the linearization methods shows, that in the moment the tracked objects are very close to each other, the measurement linearization is slightly better. Since the measurement linearization is assumed to be an approximation, this result is unexpected. Actually, the linearization at the position of the state is the approximation, not the one at the converted measurement. Therefore, the accuracy of the linearization, in scenarios where the measurement is convertible to the state space, should be expected to be more accurate. The reason for that is, that the linearization at the position of the measurement leads to:

$$\boldsymbol{z}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_{k+1|k}) = \boldsymbol{z}_k - \boldsymbol{z}_k = 0 \tag{3.36}$$

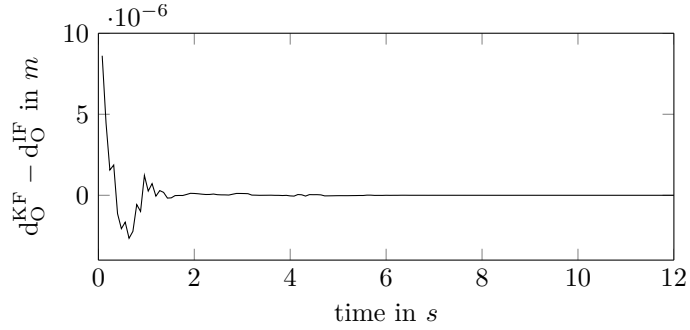
$$\boldsymbol{\nu}_k + \boldsymbol{h}_{\boldsymbol{x},k+1}\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{z}_k \tag{3.37}$$

$$\boldsymbol{i}_k = \boldsymbol{h}_{\boldsymbol{x},k+1}^{\mathrm{T}}\boldsymbol{R}_k^{-1}\boldsymbol{h}_{\boldsymbol{x},k+1}\boldsymbol{z}_k \tag{3.38}$$

and therefore to a linear measurement model as in Equation (2.29).

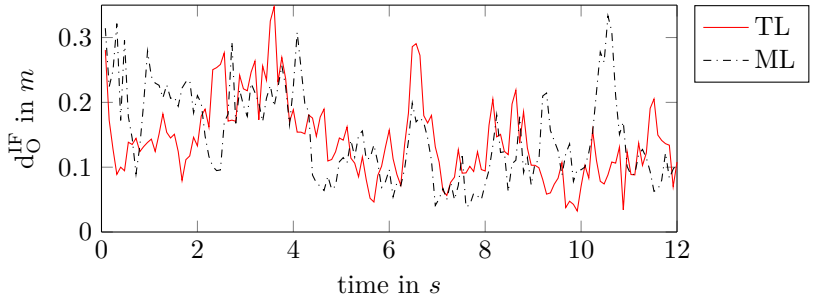


(a) OSPA distance of the information filter.

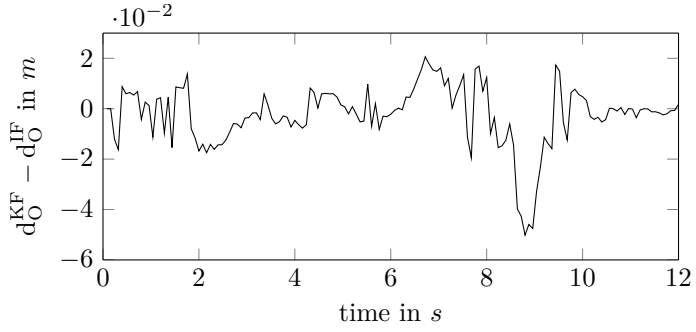


(b) Difference of the OSPA distances in state and information space.

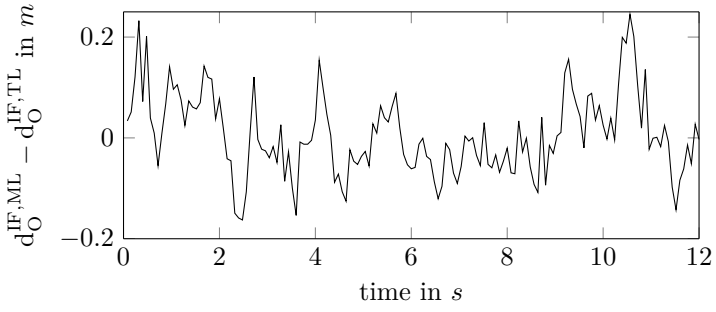
Figure 3.4: Evaluation of the OSPA distances in the simulation scenario. The used measurement model is linear.



(a) OSPAT distance of the information filter. The red solid line shows the result for linearization of the measurement model at the predicted state (TL), and the black dash dotted line at the measurement transformed to the state space (ML).



(b) Difference of the OSPAT distances in state and information space. The measurement model was linearized at the predicted state (TL).



(c) Difference of the OSPAT distances of the TL and the ML linearization approach.

Figure 3.5: Evaluation of the OSPAT distances in the simulation scenario. The used measurement model is non-linear.

3.4.2 Real-Data Results

In the real-data scenario several pedestrians are tracked indoors. The pedestrians do not have any preferred direction of moving and get very close to each other. This results in frequent partial and full occlusions. The measurements of the objects are obtained using two light detection and ranging (LiDAR) sensors, where the data of both is combined in a segmentation process presented in [RD09]. The used process model is a constant velocity (CV) model with the state vector $\boldsymbol{x} = [x\ v_x\ y\ v_y]^{\mathrm{T}}$. Therein, x and y are the Cartesian coordinates, v_x and v_y are the respective velocities. For details refer to Section 4.4.1. The measurement model was chosen to be linear: $\boldsymbol{z} = [x\ y]^{\mathrm{T}}$. The reference data for this scenario was obtained by labeling the sequence manually using the sensors' raw data. The chosen tracking parameters are given in Table 3.3. A snippet of the trajectories of the reference is shown in Figure 3.6.

Part	Parameters
Process	$\sigma_{vx} = 2.5\frac{m}{s}$; $\sigma_{vy} = 2.5\frac{m}{s}$
Tracker	$p_D = 0.8$; $p_G = 0.95$; $p_P = 0.99$; $p_{\textcircled{b},prior} = 0.2$ track is born if: $p_{\textcircled{b}} > 0.5$ track is killed if: $p_{k\|k,\exists} < 0.2$
Sensor	$\sigma_x = 0.07m$; $\sigma_y = 0.07m$
OSPAT	$\mathfrak{a} = 10$; $\mathfrak{c} = 10$; $\mathfrak{p} = 1$

Table 3.3: Tracking parameters for the evaluation of the linear and non-linear measurement model in the real-data scenario.

As before, the results in Figure 3.7 show that there is hardly any difference in the OSPAT distances. The errors in the range of 10^{-9} are again assumed to be caused by computational inaccuracies. Therefore, the results of the real-data scenario are consistent with the result of the simulated scenario.

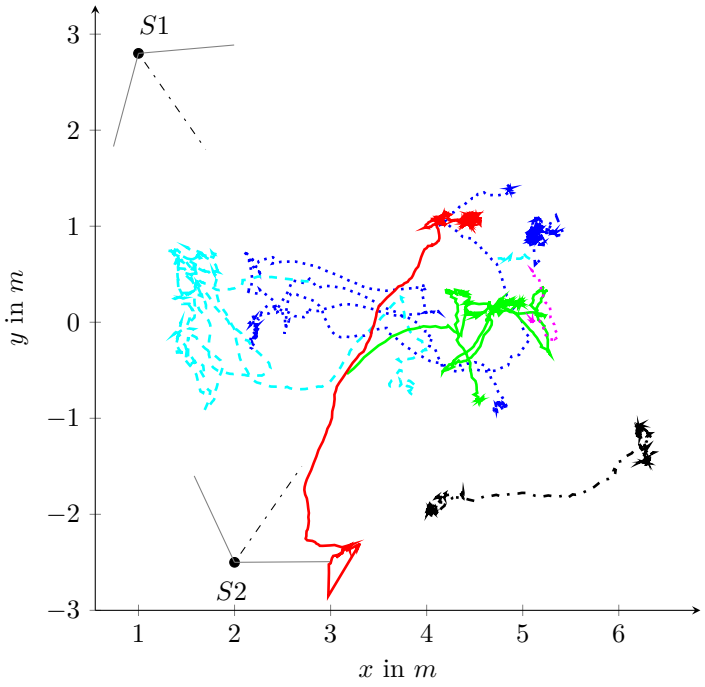
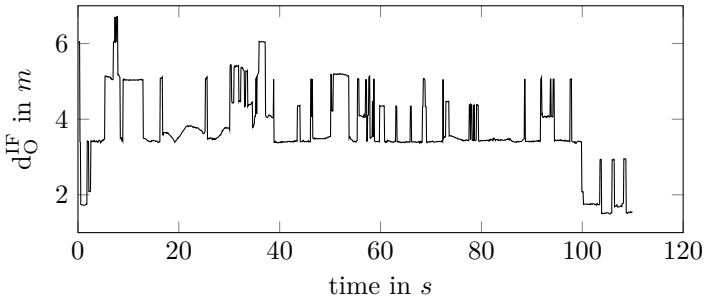
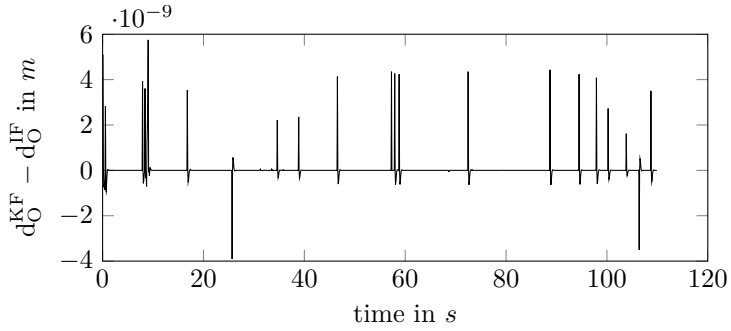


Figure 3.6: Reference trajectories of the objects in the indoor scenario in time interval $[48s, 96s]$. The black dots $S1$ and $S2$ are the position of the used sensors. The black dash dotted lines of the sensors are the sensor yaw angles and the gray lines represent the field of view. All other lines depict trajectories from different objects.



(a) OSPAT distance of the information filter.



(b) Difference of the OSPAT distances in state and information space.

Figure 3.7: Evaluation of the OSPAT distances in the real-data scenario. The used measurement model is linear.

3.5 Summary

In this chapter the JIPDA in information space was presented and evaluated. Additionally, a solution to use the information space approach in many other fusion systems, which are based on Gaussian measurement likelihoods, was given. The evaluation of the JIPDA approach shows that, in the case of linear measurement models, the information space is mathematically equivalent to the state space, except for computational inaccuracies. In case of non-linear measurement models, the approximation errors due to the use of pseudo-determinants and pseudo-inverses are not negligible but still acceptable. In the case of strong non-linearities the expected error is difficult to estimate in advance and needs to be evaluated in detail for the specific case. In addition, with the use of a non-linear measurement model, a backward channel might be necessary if the linearization at the state of the measurement is not feasible. The disadvantages lie in the higher amount of data that has to be transmitted in case of small measurement vectors and the fact that the content of a measurement itself is not interpretable. The main advantages of this approach are:

- The information measurement of every sensor has the same dimension for every type of measurement. This allows a common sensor interface independent of the measurement principle. In contrast to that, the dimension of the state space measurement may be different for every sensor.

- A side effect of the constant dimension of the measurement in information space is that the amount of values to transmit with every measurement cycle only depends on the dimension of the state vector n_x as depicted in Table 3.1.

- The fusion framework does not need any knowledge about the sensor, since all information such as the measurement model and the measurement noise are incorporated in the information measurement.

- It is impossible to get detailed knowledge about the sensor and its measurement model using the information measurement.

In summary, the presented information filter approach allows the anonymization of the sensor and leads to the associated advantages at the expense of almost negligible errors.

Chapter 4

New Approach to Processing Dense Sensor Data

Modern sensors with the ability to provide distance measures in a very high density, further referred to as high density distance measuring (HDDM) sensors, are getting more and more popular. Current state of the art sensors providing dense data are e.g. stereo cameras or LiDAR sensors. In future, high resolution radar and even better LiDAR sensors will find their way into everyday life. Examples are the existing autonomous vehicle presented in [KNW+15] or the well known Waymo project [Wik17]. Such sophisticated sensors share one property: they supply three dimensional data as point clouds. The ability of humans to analyze such point clouds is tremendous. This is the reason why one could assume that it is very easy to extract object hypotheses from this data, but humans have the ability to learn such things with a huge amount of training data, to use information about the context, and eventually fuse the perception with other sensor data (e.g. a two dimensional image of the scene). Since the fusion with other sensors in the preprocessing of a sensor is unrewarding and learning respectively classifying data can be very costly in terms of time and resources, most approaches to processing dense data are frame based. Such frame based methods are usually based on fitting or clustering. In this chapter, a new approach to extract measurements from HDDM sensors by tracking the sensor raw data and dynamically clustering the resulting tracks is presented. After explaining sensor requirements on generic data fusion in general, the sensor setup used in this work is introduced. Subsequently, the new approach is depicted in detail.

4.1 State of the Art

In order to generate measurement hypotheses from point clouds the most common approach is segmenting the data into clusters and then trying to extract properties like position, size, and yaw angle. The clustering itself is a widespread task and there are many different algorithms to solve it. One very common approach is the density-based spatial clustering of applications with noise (DBSCAN) [EKSX96] algorithm because of its simple implementation, its efficiency, and the fact that no prior knowledge about the number of clusters is necessary. Once a cluster is found, there are different possibilities to generate the hypotheses. A popular way to do this is a model based fitting of the data. In this work a box fit algorithm, further referred to as BoxFit, as described in [MDM09] is used as a reference. This algorithm, as depicted in Figure 4.1, uses a combination of the iterative end point fit (IEPF) algorithm [Ram72], a least squares line fit, and a conclusive box fit. This algorithm is specialized to detect vehicles, especially from behind when the cluster is a clean L or I shape. Figure 4.2 makes clear that generating hypotheses without knowledge about the context is not always an easy task. After exposing the scenario in Figure 4.2 to be a roundabout the scene might get clearer. Revealing one car right in front at $[x, y] = [10m, 0m]$ crossing from left to right, one car leaving the roundabout at $[20m, -18m]$, and a bus in the roundabout at $[40m, -10m]$ the scene could be regarded as obvious. The problem of frame based approaches is that every frame is independent from the last one. So the algorithm has to interpret the scene in every time step all over again. Another problem is the absence of clean I and L shapes, as used in [MDM09], if the data is accumulated over multiple sensors or multiple layers as depicted in Figure 4.3. Therefore, the fitting is done for every sensor and for every layer separately. This leads to a massive reduction of the amount of available data points and causes unsatisfying detection rates in higher distances. Further, it takes a lot of effort to estimate the best set for the necessary heuristic parameters. The parameter set strongly depends on the scenario and on the object class to perceive. A change from e.g. rural roads to a city environment can cause an immense performance drop. The drawbacks listed here are a particular problem of the used box fitting algorithm from [MDM09] but at the same time the problems are common to many clustering and fitting approaches.
Another frame based approach to deal with multiple measurements per object is to model the object's extension. Such an approach was, among others, presented in [FF08], where the extension of an object is modeled as a random symmetric positive definite matrix, with the drawback of an elliptical shape. To overcome the problems of a frame based measurement hypotheses generation, different approaches are conceivable. If available, more knowledge about the context could be used to improve the generated hypotheses. One example for that is to assume the measurements hypotheses to be orientated along the road [Mei16]. On the one hand this would improve the yaw angle estimation notably. On the other hand it might

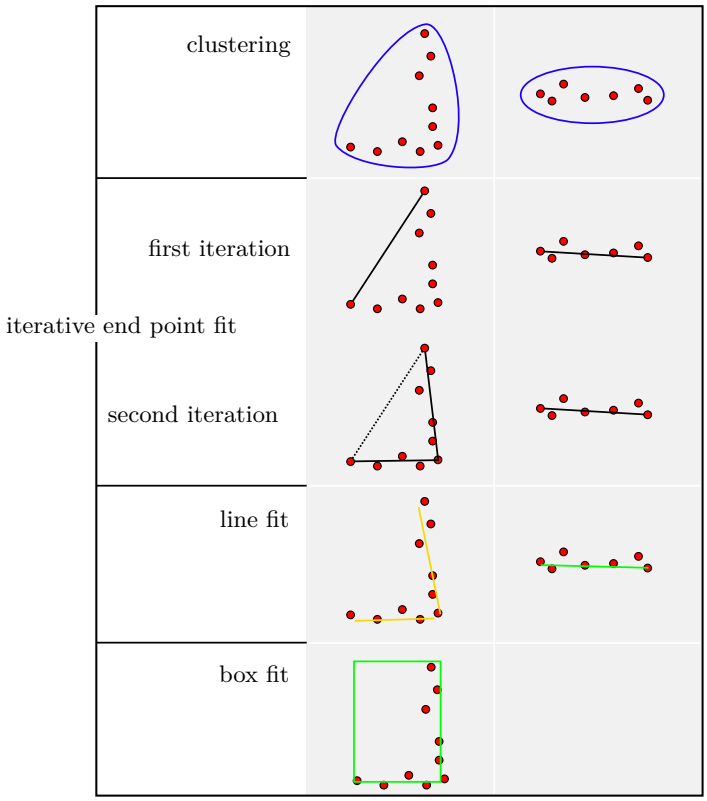


Figure 4.1: BoxFit algorithm from [MDM09] as a representative for model based clustering and fitting. After clustering the point cloud an iterative end point fit [Ram72] with a maximum of two iterations determines the possible line end points. A least squares line fit is done using the extracted end points and in case of two lines a box fitting algorithm is applied. The result of the algorithm is a line for I shapes and a box for L shapes. A side product is an estimation for the yaw angle of the clustered object. The algorithm is highly adapted to vehicles and heuristics are inevitable.

be a more proper way to incorporate knowledge about the environment in a later processing step when all available information is fused to an environmental model as presented in [NSD14]. A more elegant approach could be to imitate the human behavior to analyze the scene over time. This leads to a filtering approach, where the raw data is used as an input to a tracking. Tracking road users with raw data and

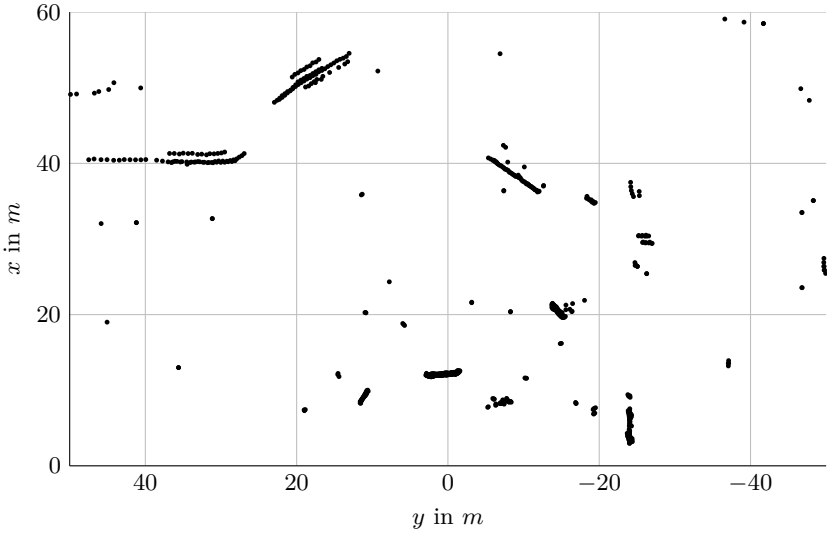


Figure 4.2: Bird's-eye view of example sensor data. For the explanation of the scenario refer to the text in this section.

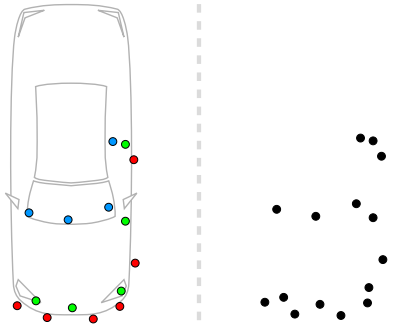


Figure 4.3: On the left: measurements from multiple layers or sensors depicted in different colors. On the right: resulting cluster of all points without any clear L or I shape as expected in [MDM09].

eventually fusing other sensors could be done with filter techniques like the group tracking using the probability hypothesis density (PHD) filter [EAGG13] and the extended target PHD tracking [GLO10; SC12]. Recently Scheel et al. presented a laser based extended object tracking using random finite set (RFS) in [SRD16]. The idea of the extended object tracking is, to model that an object can produce more than one measurement and therefore, an object can be represented by a complete

point cloud. These approaches are very promising but computationally expensive. A few years ago, Meissner et al. presented a promising clustering algorithm at an intersection, where the clusters are determined over time using measurement grids [MRD13]. This is a smart possibility to generate object hypotheses in a static environment. In a dynamic environment a static measurement grid needs to be extended by a dynamic component as well and would become a static occupancy grid as presented in [TBF05]. Such an occupancy grid is normally meant to deal with static objects. Handling dynamic objects in static occupancy grids was done, among others, in [YNK+15] and is referred to as a dynamic occupancy grid. Using dynamic occupancy grids extends the geometric space of static occupancy grids by the time domain. From a statistical point of view it is a single target tracker on singe cell basis. In comparison to the here presented approach, the output of a grid map is not only detected objects. It is more an environmental model representation. The idea of this work is to create measurement hypotheses in a generic way without the assumption of any model in a highly dynamic environment also providing additional information like yaw angle and velocity. The approach presented here is to filter the sensor raw data itself. Therefore every measurement point is regarded as a single object to be tracked. In comparison to former clustering and fitting approaches this allows to use additional information like yaw angle, velocity, and the track ID in the clustering step. Details on the new approach are presented in the following sections.

4.2 Sensor Requirements

To be able to fuse data from a sensor in a probabilistic manner, a lot of information from the sensor is necessary. The most important information obviously is the measurement $\boldsymbol{z}$ itself. A Kalman filter based fusion framework, as introduced in Section 2.1, furthermore relies on the measurement matrix $\boldsymbol{H}$, and therefore on the measurement model, and the measurement noise $\boldsymbol{R}$. In addition, as shown in Section 2.3.2, most probabilistic approaches require information like the detection probability p_D, the clutter probability p_F, and the birth probability $p_{\textcircled{b}}$. Sometimes, information about the field of view of a sensor is also useful. Since all these additional information might be dynamic, they have to be transmitted constantly.
Using the information filter as presented in Section 2.2, the information about the measurement, its model and its noise are contained in $\boldsymbol{i}$, $\boldsymbol{I}$, $|\boldsymbol{H}|_+$, and $|\boldsymbol{R}|$. As already elucidated, it is best to use linear measurement models, when using the information filter in data fusion to reduce the errors inheriting from the pseudo-inverse and pseudo-determinant. In case of distance measuring sensors, a conversion from a non-linear to a linear measurement model in the sensor is rather simple. Data from a stereo camera can be represented in Cartesian coordinates as well and therefore

a linear measurement model can be used. Using highly non-linear sensors, e.g. projecting sensors like cameras, a conversion to Cartesian coordinates, and therefore a linear measurement model, is not possible without additional assumptions. In ADAS, the most commonly tracked states are: position, yaw angle, velocity, and the respective derivatives. Therefore, a linear measurement model should provide some of these states. HDDM sensors are usually able to provide the position of an object. When using modern high resolution radar sensors, the measurement of the velocity is also possible even though it is only a radial velocity. Measuring the yaw angle (orientation) directly, is not possible in most cases but would be a very useful information. A more exact information about the velocity would be useful as well, especially in the initialization process of a new track.

4.3 Sensor Setup

The work of this chapter was done using three IBEO LUX 4 LiDARs [Ibe13] mounted at the front of an experimental vehicle as shown in Figure 4.4. Technical facts and

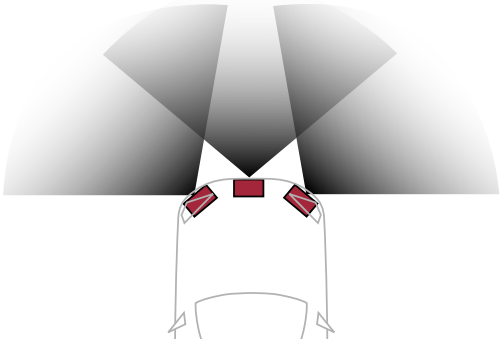


Figure 4.4: Sensor setup of the experimental vehicle: three LiDARs, depicted as red rectangles, are mounted at the front of the vehicle. The opening angle of each sensor is visualized with a black cone. (The cone does *not* show the maximum observation distance.)

the used configuration of the sensors is given in Table 4.1. Using the extrinsic calibration of the sensors, the data of all three sensors can be transformed into the vehicle coordinate system shown in Figure 4.5. An example measurement using all three sensors is depicted in a bird's-eye view in Figure 4.2. One drawback of a rotating sensor principle all LiDARs share, is that not all measurements are obtained at exactly the same time. Having the time stamp for every single measurement point,

horizontal field of view	in two central layers: 110° (50° to $-60°$) top & bottom layer: 85° (35° to $-50°$)
vertical field of view	3.2° in four layers
data update rate	set to $25.0Hz$ (also possible: $12.5Hz$ and $50.0Hz$ with reduced amount of data)
angular resolution	*horizontal*: set to 0.25° (also possible: 0.125° in central measurement area) *vertical*: 0.8°
measurement principle	time of flight
time per scan	$0.0122s$ at $25.0Hz$ and 110° field of view.
accuracy	$0.10m$

Table 4.1: Technical facts [Ibe13] and configuration of the used LUX4 LiDARs

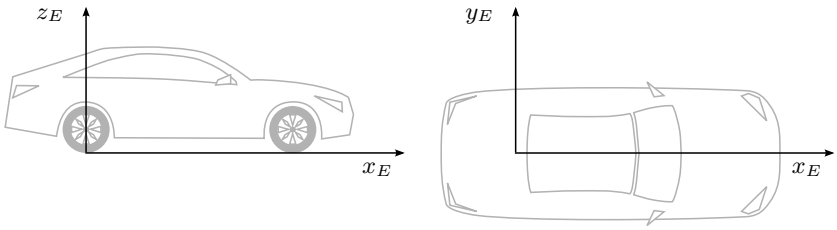


Figure 4.5: Vehicle coordinate system conforming the norm DIN 70000 which is equivalent to the norm ISO 8855.

this could be taken into account. In this work, this effect was ignored for two reasons: the problem is not that important as Table 4.2 points out. For an object crossing the sensor field of view orthogonally, this effect is only $4cm$ in a distance of $10m$ at a speed of $50\frac{km}{h}$. With a given accuracy of $10cm$ for the sensor, this is negligible. Further, as soon as the measurements of one or more sensors are used in the same coordinate system for any reason, e.g. to apply clustering algorithms, it has to be assumed that all measurements were obtained at the same time. Without exact knowledge about the object and its moving direction the knowledge about the time stamp can not be used for corrections. So far, the problems occur only for the objects' extent and movement orthogonally to the laser ray direction. The perpendicular extent and movement of objects do not cause such problems. As soon as the sensor is moving perpendicularly to its laser ray direction a corresponding error occurs. For measurements from the same sensor in two successive scans the resulting movement

is corrected by using the ego motion as depicted later in Section 4.4.1.

Distance of object in $[m]$		2.0	10.0	50.0	100.0
Δy in $[m]$	at $v = 50\frac{km}{h}$	$0,158$	$0,043$	$0,009$	$0,004$
	at $v = 100\frac{km}{h}$	$0,317$	$0,087$	$0,018$	$0,009$

Table 4.2: Distance Δy an object covers in the time of one scan when crossing the sensor field of view orthogonally depending on the distance and the velocity of the object. Assumed length of the object is $5m$. Sensor parameters are as listed in Table 4.1.

4.4 Hypotheses Generation using Raw Data Tracking

The preprocessing method for HDDM sensor data in this work aims for generating hypotheses for surrounding extended objects without using any assumptions about the type of object. Such an hypothesis $\mathcal{H}$ with:

$$\mathcal{H} = \{\mathfrak{h} = [x_\mathcal{H}, y_\mathcal{H}, v_\mathcal{H}, \psi_\mathcal{H}]^\mathrm{T}; \boldsymbol{P}_\mathfrak{h}; [l_\mathcal{H}, w_\mathcal{H}]^\mathrm{T}; \mathcal{RP}_\mathcal{H}\} \tag{4.1}$$

consists of the hypothesis state vector $\mathfrak{h} = [x_\mathcal{H}, y_\mathcal{H}, v_\mathcal{H}, \psi_\mathcal{H}]^\mathrm{T}$, with the two dimensional position $[x_\mathcal{H}, y_\mathcal{H}]^\mathrm{T}$, the velocity $v_\mathcal{H}$, and the yaw angle (orientation) $\psi_\mathcal{H}$. Further, $\boldsymbol{P}_\mathfrak{h}$ denotes the covariance matrix for the vector $\mathfrak{h}$. An extent for the hypothesis is given with length $l_\mathcal{H}$ and width $w_\mathcal{H}$. For an exact positioning in subsequent algorithms the reference point $\mathcal{RP}_\mathcal{H}$, which describes where on the extended objects the given position is valid, is given. More information about reference points is given later in this section. A common clustering approach incorporating only information from a single time step is not able to generate reliable propositions about the angle and has problems in scenarios where multiple moving and/or non moving objects are close to each other. The basic idea of the approach presented in this work is to filter the raw sensor data over time and to use the gained information to generate reliable hypotheses for the perceived objects. Therefore, every measurement point is estimated over time using one instance of a multi-instance Kalman filter. In the following, the filtered position of the measurements are called raw data tracks or just tracks. Figure 4.6 shows the information flow of the raw data tracking approach. The dense point cloud is the input and the object hypotheses $\left[\mathcal{H}^1\ \mathcal{H}^2 \cdots \mathcal{H}^n\right]$ are the output. These can be seen as the new raw data based measurements of the sensor after the preprocessing and can be used in a more sophisticated subsequent object tracking algorithm. The chart in Figure 4.6 shows that the result of the clustering

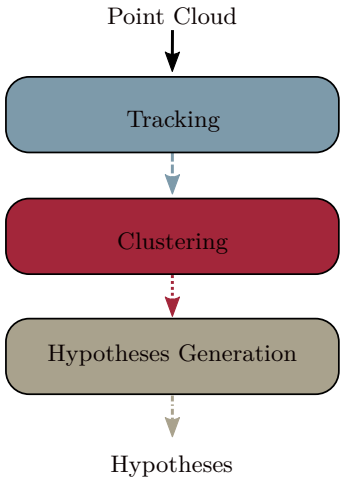


Figure 4.6: Flow chart of the raw data tracking. There are three steps: *tracking* the raw data (Section 4.4.1), *clustering* the tracks (Section 4.4.2), and the *hypotheses generation* (Section 4.4.3). The result of the hypotheses generation is not reused in the algorithm and does not influence the tracking.

and hypotheses generation does not influence the tracking part of the algorithm. This increases the generality of the algorithm, since each of the three parts is exchangeable. Further, it is possible to reuse the results of the computationally rather expensive tracking part in multiple clustering methods. In the course of this section, the three parts of the algorithm depicted in Figure 4.6 are explained in detail.

4.4.1 Tracking Raw Data

The main idea of the preprocessing algorithm for HDDM sensors presented in this work is to treat every measurement of the dense input point cloud as an independent object and to track track it with a multi-instance Kalman filter. Details on the implemented tracking algorithm are subject of this section.

Process and Measurement Model

In every Kalman filter a model for the behavior of objects, the process model, and for the measurements, the measurement model, is needed. The preprocessing algorithm

in this work is designed to work with any kind of moving object without or only small adaption of the model's parameters. The objects to be tracked are single points of a point cloud without any knowledge about which kind of extended object has caused the measurement point. Thus, the behavior of a measurement point needs to be modeled. Since the goal of the preprocessing is to generate hypotheses for moving and non moving objects, where these two kinds of objects are not separable from each other without additional information, the best assumption about a model covering nearly every kind of movement is: any tracked point can move in any direction at any time under the constraint of a continuous movement. Therein continuous movement can be modeled with the assumption of a constant velocity. Even pedestrians who are able to change the main direction of moving instantaneously, prefer a continuous movement in most cases. A two dimensional CV model is used as an approximation of such a behavior, where a change of velocity and moving direction is modeled by a constant velocity with an added acceleration noise. Actually, the best approach for static objects seems to be a constant position model, but this is insufficient for tracking moving objects. The CV model can handle static objects as well and is able to deal with the transition from static to moving. It is also known as white noise acceleration model and was derived for one dimension in [BL93]. Assuming that the movements in x and y are independent from each other, the two dimensional model used here is the concatenation of the equations of two one dimensional models from [BL93]. This results in the state vector:

$$\hat{\boldsymbol{x}}_k = \begin{bmatrix} x \\ v_x \\ y \\ v_y \end{bmatrix}, \tag{4.2}$$

where $[x\ y]$ is the two dimensional position and $[v_x\ v_y]$ the corresponding velocity vector. The process model of the object is given by the state transition matrix:

$$\boldsymbol{F}_k = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

with the time difference T between two consecutive measurement cycles. As one can see here, the velocity does not change over time and is therefore constant. The last

part of the process model is the process noise matrix:

$$\boldsymbol{Q}_k = \begin{bmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 & 0 & 0 \\ \frac{1}{2}T^3 & T^2 & 0 & 0 \\ 0 & 0 & \frac{1}{4}T^4 & \frac{1}{2}T^3 \\ 0 & 0 & \frac{1}{2}T^3 & T^2 \end{bmatrix} \begin{bmatrix} \sigma_{v_x}^2 & 0 & 0 & 0 \\ 0 & \sigma_{v_x}^2 & 0 & 0 \\ 0 & 0 & \sigma_{v_y}^2 & 0 \\ 0 & 0 & 0 & \sigma_{v_y}^2 \end{bmatrix}. \tag{4.4}$$

Therein, $[\sigma_{v_x}\ \sigma_{v_y}]$ are the acceleration noises. Here, it is not very reasonable to have different noise parameters in different directions, but for other applications this might be helpful. E.g. a static setup at a sidewalk, where it is more probable, that pedestrians move along the sidewalk than crossing it. In this case the lateral noise could be smaller than the longitudinal.
As already mentioned, the input to the raw data tracking are Cartesian measurements:

$$\boldsymbol{z}_k = \begin{bmatrix} x_{LS} \\ y_{LS} \end{bmatrix}. \tag{4.5}$$

This leads to a linear measurement model with the measurement matrix:

$$\boldsymbol{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.6}$$

For the measurement covariance matrix $\boldsymbol{R}$ from Equation (2.4) it follows:

$$\boldsymbol{R}_k = \begin{bmatrix} \sigma_{x_{LS}}^2 & \sigma_{x_{LS}}\sigma_{y_{LS}} \\ \sigma_{x_{LS}}\sigma_{y_{LS}} & \sigma_{y_{LS}}^2 \end{bmatrix}, \tag{4.7}$$

with the measurement noise $\sigma_{x_{LS}}$ in x direction and $\sigma_{y_{LS}}$ in y direction. Here, it is assumed that the measurement noise in x and y are uncorrelated:

$$\boldsymbol{R}_k = \begin{bmatrix} \sigma_{x_{LS}}^2 & 0 \\ 0 & \sigma_{y_{LS}}^2 \end{bmatrix}. \tag{4.8}$$

Algorithm Overview

The input to the tracking are the measurement points of the sensor in Cartesian coordinates $[x_{LS}\ y_{LS}]^{\mathrm{T}}$. The third dimension is not necessary and not using it reduces the computational load significantly. All points from the different sensors are transformed into a common coordinate system, the vehicle coordinates as depicted in Figure 4.5, and are projected onto the ground plane (by setting the third dimension to zero). There may be multiple measurements at approximately the same position when using multiple sensors with overlapping fields of view. In order to reduce

the complexity and the computational load, a discretization is done. Sophisticated tracking algorithms, like RFS, usually are computationally too expensive to handle such an amount of tracks given here in between two measurement cycles of typical sensors. For the same reason probabilistic approaches like PDA or multi-hypothesis tracking (MHT) were not chosen. Therefore, the rather simple but effective method of nearest neighbor association is used to solve the measurement-to-track association. Using such a low level method has known drawbacks like frequent track losses because of wrong associations, but in this case a perfect match from measurements to tracks is not possible and not needed. It does not matter if close tracks caused by the same extended object switch measurements as long as the corresponding track cloud moves like a swarm. The output of the algorithm are the raw data tracks which are used in the subsequent hypotheses generation as depicted in Figure 4.7. Details on the elements of the flow chart in Figure 4.7 are subject of the following sections.

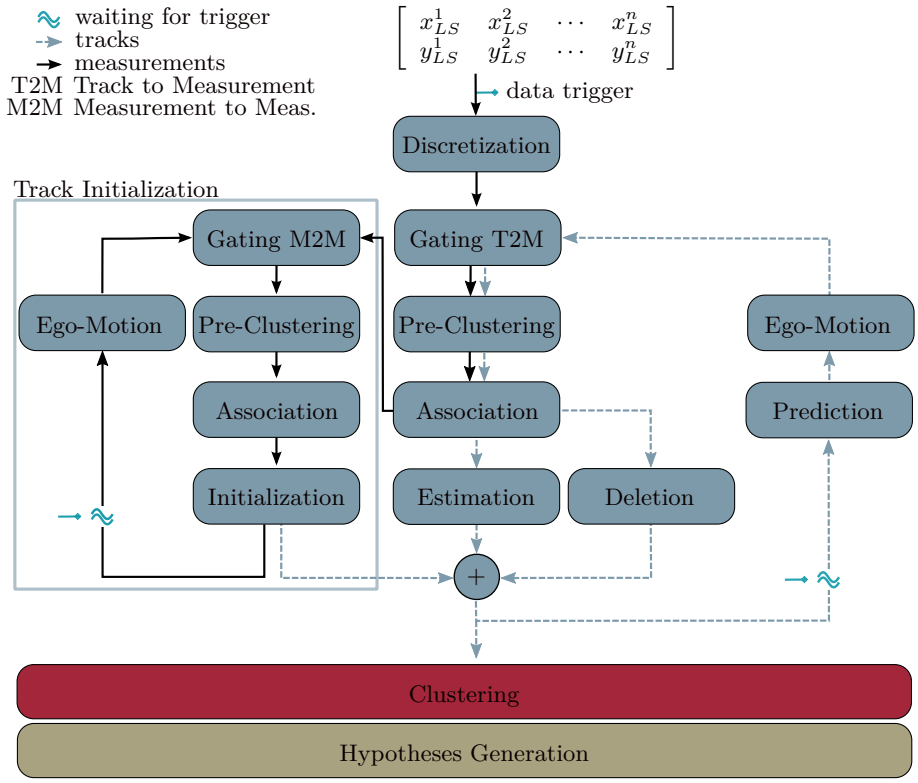


Figure 4.7: Flow chart of the raw data tracking algorithm. Input to the algorithm is a point cloud, output for the subsequent clustering are the raw data tracks. The data flow for the point cloud (measurements) is depicted as solid black arrow and for the tracks as dashed gray arrows. The box on the left marks the part of the algorithm which is used to initialize new tracks.

Discretization

In order to handle the overlap of the sensors' fields of view and to reduce the computational load the point cloud is discretized. This is done using a measurement grid, where all measurements are inserted to. When choosing the discretization size of the grid d_{grid} in the range of the sensor accuracy, the effective inaccuracy of the system is dominated by the latter and hardly changed by the discretization. Thus, loss of information can be assumed to be small. After inserting all measurements from all sensors into the grid, the middle of every occupied cell, further referred to

as the grid measurement, is used as input to the tracking. Calculating a mean or median value inside of every cell would create a unnecessary computational overhead since it is very likely that the points are spread over the whole cell if the grid size is comparable to the sensor accuracy as shown in Figure 4.8. Note that the

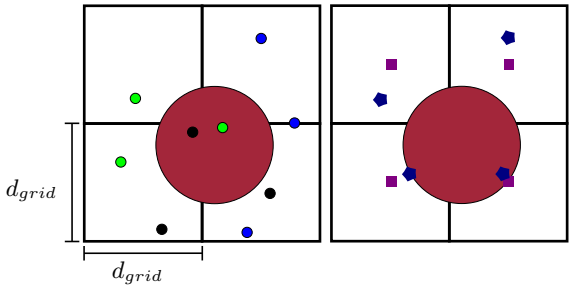


Figure 4.8: Effects of discretization grid: black, green, and blue dots are measurements. The big red circles are objects and the black squares are the grid. The purple squares denote the middle of an occupied cell and the blue hexagons represent the mean values of the measurements within the cell. The grid size d_{grid} is chosen to be approximately the same as the sensor accuracy. Thus, the measurements are typically spread over the whole cell. The loss of information due to the discretization is therefore hardly noticeable and the advantage of calculating the mean value of the data points over using the center of the cell is purely random.

discretization is part of the measurement acquisition process. The tracking and the tracks themselves are not subject to any discretization. After the discretization, there may be still several hundred grid measurements per time step.

Gating Track to Measurement

Gating is a method to reduce the amount of possible association solutions between tracks and measurements. Without the gating, the association algorithm has to consider all grid measurements and all available raw data tracks. This can be quite demanding and time-consuming. In the gating step it is evaluated if there are track-to-measurement relations which are very unlikely to be associated. Therefor it is necessary to calculate a distance measure between a raw data track and a grid measurement. As already pointed out, the MHD from Equation (2.33) can be used as such a measure. It is calculated between the predicted track and the grid measurement with an assumed measurement noise. Therein, the minimum

of the measurement noise should be the accuracy of the sensor or, if greater, half the size of the discretization grid size: $d_{grid}/2$. Since the calculation of the MHD is computational expensive, a fast gating is done in advance. Before calculating the complex MHD, it is checked if the track and the measurement exceed a distance threshold and are too far away from each other in either the x or the y direction. Only if the association passes the fast gating check the more complex MHD is calculated. The output of the gating is a boolean value: it is true if the association is possible. At the same time the MHD is saved, since it is used again in the association step.

Gating Measurement to Measurement

The gating from unassociated grid measurements from the previous time step to current measurements differs from the gating of tracks to measurements, since there is no information about the movement of the possible object. Thus, it is not possible to predict the measurements from the previous time step to the current one. Only the ego motion correction is applicable. An association basing only on the measurement noise using the MHD is not sufficient in case of a moving object. The only possibility to exclude any measurement to measurement association in the gating is the spatial position. Using an assumption about the maximum velocity of an object, all possible associations have to be inside of circle around a measurement defined by the maximum covered distance. The output of the gating is again a boolean value saying if the association is possible or not.

Pre-Clustering

The pre-clustering is a further step to reduce the complexity of the association problem. Since it is done twice, in the normal and in the track initialization branch in Figure 4.7, there are two possible types of input: tracks and unassociated grid measurements from the previous time step. In the following both are called candidates. The idea of the pre-clustering is to use the boolean output from the gating to check if there is any candidate sharing a possible grid measurement association with another. If a candidate shares at least one measurement with another, they define a cluster. If any candidate of a cluster shares a further possible measurement with any other candidate, this is added to the cluster as well. In the following association step, it is only necessary to solve the association problem for each cluster. This is helpful, since normally the computational effort increases exponentially with an increasing number of candidates but only linearly with an increasing number of clusters. Additionally, the solution is independent for every cluster and can be calculated in parallel. An

algorithm to calculate the clusters based on the gating result was presented in [DB93] and is quite common in tracking, especially in the PDA algorithm.

Association

In this step, the final solution of the association problem is calculated. As already mentioned, it is not necessary for the presented approach to have a perfect association match. It is more important to match inside the right cluster and, first of all, to have a computationally inexpensive solution. Therefore, the Hungarian method [Kuh55], also known as Munkres algorithm, is used to solve the association problem. An alternative solution would be the known auction algorithm [Ber88]. Using such a global nearest neighbor approach inside every cluster from the pre-clustering step can be quite time-consuming for large clusters. In the initialization phase of the algorithm where there are a lot of old and current measurements the clusters are very large. The same holds true for the following settling phase of the algorithm where the track uncertainties are quite high and the resulting clusters are very large and quite often end up in one huge cluster. Something similar might happen if many objects get very close to each other and behave similar, e.g. are standing still. In such cases, the computational effort has to be reduced. This is done by approximating the global nearest neighbor algorithm with a local nearest neighbor solution which has only quadratic complexity in comparison to the cubic complexity (best case, depending on the implementation) of the Hungarian method. In case of a small amount of data this difference is not really relevant, but f.i. having 400 tracks and measurements to associate, the difference from square ($400 \cdot 400 = 160.000$) to cubic ($400 \cdot 400 \cdot 400 = 64.000.000$) is 63.840.000 operations less using a local nearest neighbor approach. It is obvious, that using the local approach results in non perfect matches in the association, but it allows to continue the algorithm using a reduced solution until the cluster size is small enough. This happens always right after starting the algorithm. After a few time steps the tracks are good enough to form smaller clusters and a global nearest neighbor algorithm is used.

Estimation

After determining a unique association from grid measurements to predicted raw data tracks, the next step of the Kalman filtering is the estimation. Using the filter equations from 2.7 to 2.11 every track is estimated, also called updated, by incorporating the associated measurement. The estimated raw data tracks of this step are the main output of the tracking and are used in the subsequent hypotheses generation.

Initialization

The initialization of tracks is the concluding task of the track initialization branch in Figure 4.7. After associating measurements of the previous step with measurements of the current step, tracks can be initialized using a two step initialization as presented in [BL93]:

$$\hat{\boldsymbol{x}}_{0|0} = \begin{bmatrix} \boldsymbol{z}_0(0) \\ (\boldsymbol{z}_0(0) - z_{-1}(0))/T \\ \boldsymbol{z}_0(1) \\ (\boldsymbol{z}_0(1) - z_{-1}(0))/T \end{bmatrix} \tag{4.9}$$

$$\boldsymbol{P}_{0|0} = \begin{bmatrix} \sigma^2_{x_{LS}} & \sigma^2_{x_{LS}}/T & 0 & 0 \\ \sigma^2_{x_{LS}}/T & 2\sigma^2_{x_{LS}}/T^2 & 0 & 0 \\ 0 & 0 & \sigma^2_{y_{LS}} & \sigma^2_{y_{LS}}/T \\ 0 & 0 & \sigma^2_{y_{LS}}/T & 2\sigma^2_{y_{LS}}/T^2 \end{bmatrix} \tag{4.10}$$

Here, the same noise as in the measurement covariance matrix is used, but in implementations it is common practice to use a multiple of the noise to get a more stable initialization. If there are measurements of the current time step without an association, these measurements are saved and compensated for the ego motion. Therefore, it is necessary to wait for the subsequent time step as depicted in Figure 4.7 and explained in detail later in this section. The newly initialized tracks are, together with the estimated tracks, the main output of the tracking part which is used for hypotheses generation.

Deletion

Tracks which were not associated in the previous step are not deleted immediately. They are kept for some further time steps where the tracks are only predicted and not updated. After a certain number of predictions without an update, the track is deleted, where this number is a configuration parameter. This is an heuristic approach for track deletion. A more sophisticated approach would be to estimate an existence probability as it is done in the integrated probabilistic data association (IPDA) [MES94], JIPDA, and also in some RFS approaches. Since the here presented algorithm uses nearest neighbor algorithms, and therefore a hard unique association instead of soft association of multiple measurements, it is not possible to calculate an existence probability like in Equation (2.55). Furthermore, since there is no information about the a priori birth probability $p_{\textcircled{b},prior}$ and the persistence probability p_P, the calculation of the existence probability would not be very meaningful while increasing the computational overhead. Thus, the threshold based deletion, as

described above, is a simple and effective solution.

Prediction

After estimating and initializing all tracks, the next step of the Kalman filtering is the prediction as given in Equations (2.5) to (2.6). To be able to predict the tracks to the next time step it is necessary to know the elapsed time from one step to the other. Thus, the algorithm waits for the next data input as depicted in Figure 4.7.

Ego-Motion

After the prediction, the tracks need to be corrected because of the motion of the sensor platform, the ego vehicle. This ego motion compensation is necessary since the ego vehicle moves between two consecutive time steps and causes a virtual movement and rotation of the tracks as depicted in Figure 4.9. Using the EKF

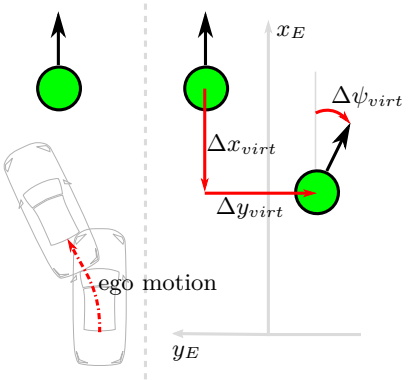


Figure 4.9: Virtual movement of a track because of the ego motion. On the left depicted in green is a track with its yaw angle and the ego vehicle. The ego vehicle moves from one time step to another while the track remains on the same position. On the right the same scenario is shown in the vehicle coordinate system $[x_E, y_E]^{\mathrm{T}}$. Because of the ego motion it seems as if the track has moved by $[\Delta x_{virt}, \Delta y_{virt}]^{\mathrm{T}}$ and turned by $\Delta\psi_{virt}$.

ego motion estimation proposed in [Mäh09], the translation and rotation of the ego vehicle is obtained. Therefore, the additional input of the vehicle velocity and

yaw rate is necessary. Further, the time difference from the current time step and the previous one is needed as depicted by the wait condition in Figure 4.7. Every track is compensated for the translation and rotation. This compensation corrects the prediction of the track in a manner that the track does not get any virtual velocities. An alternative approach is to use the control input vector $\boldsymbol{u}_k$ and the control input matrix $\boldsymbol{G}_k$ from Equation (2.5) to directly incorporate the ego motion compensation into the Kalman filter algorithm. From the implementation point of view it is advantageous to have the tracking and the ego motion compensation separately so both parts can be replaced and reused in other algorithms easily.

4.4.2 Clustering

The task of clustering is to use the output of the preceding raw data tracking (Section 4.4.1) to group tracks together which behave in a similar manner. The clusters are then input to the hypotheses generation as depicted in Figure 4.6. In comparison to the raw sensor data, the output is now smoothened using a Kalman filter and additional information in form of the velocity $v_{\mathcal{H}}$, the yaw angle $\psi_{\mathcal{H}}$, and the history of every tracked point is gained. In Figure 4.2 the raw sensor data of a roundabout was shown and the problem of the frame based object recognition was illustrated. In Figure 4.10 the same scene is shown again. This time, the green dots represent the tracked objects, and the black lines show the history of the last 40 time steps. A closer look reveals that mostly moving objects cause long histories in space and time as depicted in Figure 4.11. Spatially long but temporally short histories on not moving objects result mainly from jittering measurements causing an initialization of new tracks with high velocities like in Figure 4.12. This effect can only be removed by limiting the allowed association distance or decreasing the initial uncertainty of tracks. Since in this case an approaching vehicle with an high velocity would not be detected either, it is better to solve this problem with an outlier filtering in the clustering process. Beside an erroneous ego motion compensation, jittering of the raw measurements are the reason, why stationary objects cause raw data tracks with a velocity greater than zero.
Using the analyzing capabilities of a human, the scene in Figure 4.10 is much clearer than before. The histories of the tracks seem to be a good feature to separate real moving objects from background. However, the fact of having a spatially long history alone is not sufficient, since newly born objects have quite short temporal histories. Further, one can imagine, that the major drawback of this approach is the perception of non-moving objects. These can not be distinguished clearly from the background. When thinking about methods to cluster raw data tracks as depicted in Figure 4.10, it is reasonable to think about the methods used to cluster point clouds. The position based DBSCAN [EKSX96] is one of the most famous approaches. This algorithm can be used to find clusters basing on any symmetric distance measure, e.g. the

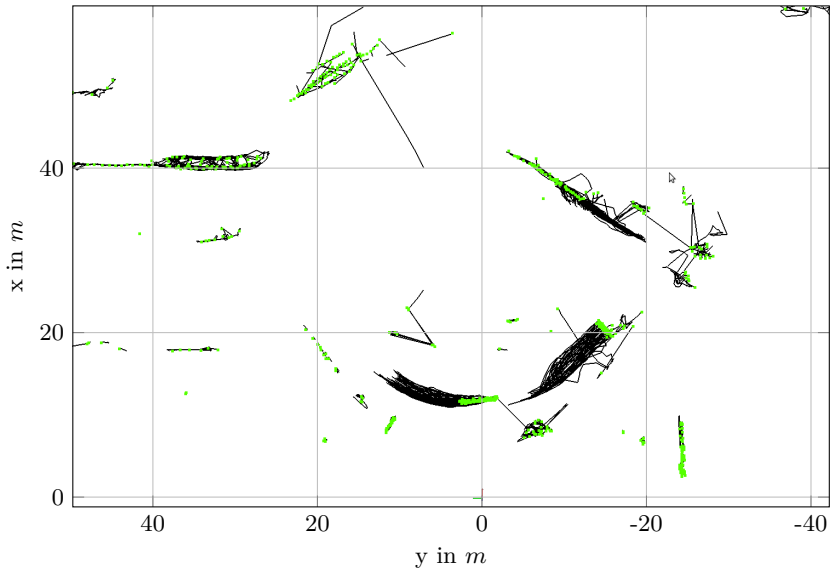
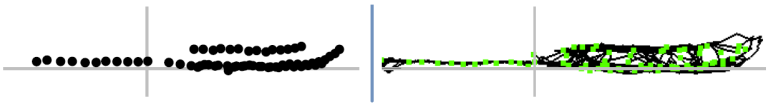
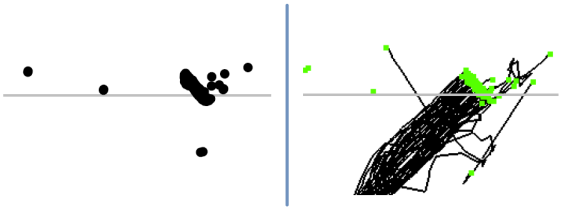


Figure 4.10: Raw data tracks in green and their histories (40 time steps) in black at the same moment as in Figure 4.2.

Euclidean distance of the filtered two dimensional positions of the raw data tracks. In this work, the DBSCAN is used as reference algorithm. Using a single frame based approach, the advantage of having a unique ID for the data to be clustered is ignored. These IDs are the result of the preceding tracking. The clustering approach in this work incorporates the IDs and was first presented in [SS06] as a graph based clustering, further called graph clustering (GC). The realization of this clustering having raw data tracks is explained in detail in this section. A graphical overview of the algorithm is given as a flow chart in Figure 4.13. The basic idea of the GC is to create clusters with a memory by adding an unique ID to the cluster. Tracks initialize a new cluster or are associated to an existing one if their distance decreases below the association distance and they are removed from the cluster if they exceed the removal distance. Detailed information on the algorithm and on the blocks in Figure 4.13 is given in the following sections.



(a) Comparison of a static object at the position $[x, y] = [40m, 40m]$. On the left side depicted with the raw sensor data taken from Figure 4.2 and on the right side with the tracked histories of the last 40 time steps taken from Figure 4.10



(b) Comparison of a moving object at the position $[x, y] = [20m, -18m]$. On the left side depicted with the raw sensor data taken from Figure 4.2 and on the right side with the tracked histories of the last 40 time steps taken from Figure 4.10

Figure 4.11: Comparison of raw sensor data and tracked histories of a static and a moving object in the same scene. The images are an cutout from Figure 4.2 and Figure 4.10

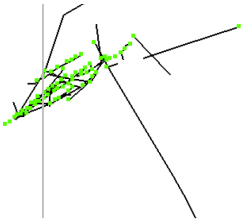


Figure 4.12: Spatially long histories on a static object caused by jittering measurements. These histories are normally only a few time steps old. The image is a cutout of Figure 4.10 at position $[x, y] = [20m, 50m]$

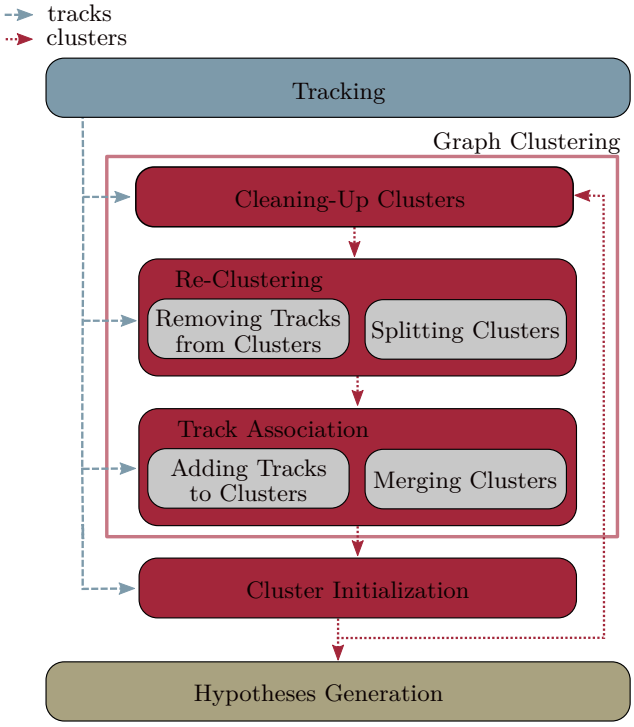


Figure 4.13: Flow chart of the graph clustering (GC) algorithm. Input are the raw data tracks and output to the hypotheses generation are the clusters. A feedback loop from the output to the cleaning-up allows to have a memory for every cluster. Disabling the feedback loop results in a frame based clustering.

Distance Measures

Like the DBSCAN, the graph clustering (GC) can be combined with any symmetric distance measure like the Euclidean distance. When clustering raw data tracks, there is more information available than only the position. There is also information about the velocity and the yaw angle, respectively the velocity vector, and even a covariance matrix providing information on the uncertainty. The idea arises to use the complete state and its covariance to do a similarity check comparable to the MHD. The drawback of the MHD is the assumption that the variances and covariances of the compared distributions are identical. Another measure of statistical similarity of two normal distributions $p_1 = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $p_2 = \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ is the Bhattacharyya distance (BD) [Bha46]:

$$d_{BD}(p_1, p_2) = \frac{1}{8}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) + \frac{1}{2} \ln \left(\frac{|\boldsymbol{\Sigma}|}{\sqrt{|\boldsymbol{\Sigma}_1||\boldsymbol{\Sigma}_2|}} \right) \tag{4.11}$$

with

$$\boldsymbol{\Sigma} = \frac{\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2}{2}. \tag{4.12}$$

In case of identical covariances $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$ the BD simplifies to:

$$d_{BD}(p_1, p_2) = \frac{1}{8}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1). \tag{4.13}$$

This is equivalent to the MHD in Equation (2.33), except for the factor $\frac{1}{8}$. The BD can be used as a distance measure in the DBSCAN and the GC. So far, the BD is an elegant way to integrate not measured states, like yaw angle and velocity, and a covariance matrix into the clustering.
Another approach to define a distance measure incorporating the velocity and yaw angle information is to define a symmetric distance d_{EM} as a weighted mixture of the Euclidean distance and other similarity measures:

$$d_{EM} = \frac{d_{xy} + \Theta(d_\psi - d_{xy}) d_\psi + \Theta(d_v - d_{xy}) d_v}{1 + \Theta(d_\psi - d_{xy}) + \Theta(d_v - d_{xy})}. \tag{4.14}$$

Therein, $\Theta(x)$ is the known Heaviside function:

$$\Theta(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}, \tag{4.15}$$

d_ψ is a difference of yaw angles, d_v is a difference of velocities, and d_{xy} is the Euclidean distance:

$$d_\psi = \alpha_\psi |\psi_1 - \psi_2| \tag{4.16}$$

$$d_v = \alpha_v |v_1 - v_2| \tag{4.17}$$

$$d_{xy} = || \left[x_1 \; y_1\right]^{\mathrm{T}} - \left[x_2 \; y_2\right]^{\mathrm{T}} ||. \tag{4.18}$$

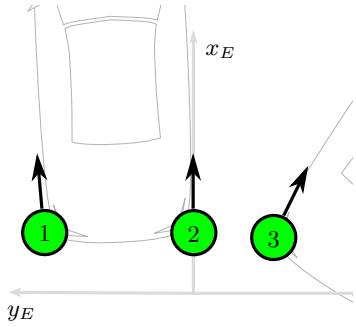
The parameters α_ψ and α_v are weighting factors. Setting the weight of the Euclidean distance d_{xy} to one, the difference of the velocity and the yaw angle need to be weighted for two reasons: the range of the distance has to be scaled to fit to the range of the Euclidean distance and to bring the differences to the same unit. This is necessary, since the range of e.g. an angle difference in radians is quite small and limited to $[0, 2\pi[$. Thus, a weight to account a difference of $5.0° \approx 0.0873 rad$ equal to a distance of $1m$ would be: $\alpha_\psi = 11.46^{m}/_{rad}$. A problem of using the angle and velocity differences without any constraint is, that tracks which are spatially far away from each other may have a small distance d_{EM} because of their similarity of the angle and the velocity. The Heaviside function in 4.14 is used to reduce this problem and to allow only an increase of the total distance by the angle and velocity distances.

Example This example shows the problem of pure position based and pure probability based distances. A problem for probabilistic distances are tracks with high uncertainties as depicted in Figure 4.14a. High uncertainties occur for example when initializing a new track. Figure 4.14a shows such a case. The states, given in $[x \; v_x \; y \; v_y]$, and its covariances are given with:

$$\hat{\boldsymbol{x}}^1_{k|k} = \begin{bmatrix} 80.0 \\ 10.0 \\ 2.0 \\ 0.875 \end{bmatrix} \qquad \hat{\boldsymbol{x}}^2_{k|k} = \begin{bmatrix} 80.0 \\ 10.0 \\ 0.0 \\ 0.0 \end{bmatrix} \qquad \hat{\boldsymbol{x}}^3_{k|k} = \begin{bmatrix} 80.0 \\ 10.0 \\ -1.0 \\ -3.64 \end{bmatrix}$$

$$\boldsymbol{P}^1_{k|k} = \mathrm{diag}\left[1, 2, 1, 2\right] \qquad \boldsymbol{P}^2_{k|k} = \mathrm{diag}\left[1, 1, 1, 1\right] \qquad \boldsymbol{P}^3_{k|k} = \mathrm{diag}\left[5, 10, 5, 10\right]$$

The corresponding distances in Figure 4.14b show that, using the Euclidean mixture, the distance for the tracks $t = 1$ and $t = 2$ is smaller than for $t = 2$ and $t = 3$ which is correct regarding the scenario. To decide if the tracks belong to the same cluster is part of the clustering process. This example is of course artificial and it is clear, that there are contrary examples, where the Euclidean mixture fails. To decide which distance shall be preferred a detailed evaluation is necessary.



(a) Example for the difficulty to calculate distances. Two vehicles drive side by side. The two tracks on the left car have a bigger Euclidean distance than the two tracks on the right which inherit from different objects.

	d_{xy}	d_{BD}	d_{EM}
$2 \rightarrow 1$	2.0	0.564	2.0
$2 \rightarrow 3$	1.0	0.343	2.5

(b) Euclidean distance d_{xy}, Bhattacharyya distance d_{BD}, and Euclidean mixture d_{EM} distance for the given scenario. For the Euclidean mixture the weighting factors are: $\alpha_\psi = 11.46$ and $\alpha_v = 0.2$

Figure 4.14: Comparison of distance measures in a difficult scenario.

Cluster Initialization

In the initialization phase a DBSCAN is used to generate clusters. The input to the cluster initialization are new tracks which were not associated to any existing cluster, but also tracks which where removed from other clusters. In the initialization the main parameter, as in most other clustering approaches, is the association distance, sometimes called connection distance. As soon as two or more tracks are closer than the association distance, the tracks initialize a new cluster, where every cluster obtains a unique cluster ID. The ID corresponds to some sort of memory, since the clusters are always fed back to the algorithm in the next time step. Disabling the feedback loop in Figure 4.13 from the output back to the cleaning-up, the GC would be exactly the same as a DBSCAN since the algorithm then has no memory and is only frame based.

Cleaning-Up

In the GC every cluster holds a list of unique track IDs given by the tracking. In every new time step it is very likely, that some of the raw data tracks are deleted for any reason, e.g they are leaving the field of view of the sensor. Thus, the clusters

need to be cleaned up by removing the deleted tracks from the corresponding cluster.

Re-Clustering

After updating the raw data tracks in a new measurement cycle, the clusters need to be re-clustered. Thus, the algorithm, as presented from [SS06], has not only an association distance which is used to associate tracks to each other and therefore to a cluster. It also has a removal distance as an additional parameter. Usually the removal distance is chosen to be bigger than the association distance to implement a hysteresis. As soon tracks are associated to each other in a cluster, their distance has to increase over the removal distance to be separated from the cluster. Basing on two distances, the re-clustering has two main tasks:

- *Removing Tracks from Clusters*
  If a track exceeds the removal distance to all tracks inside a cluster, the track is removed from the cluster and is again available for clustering in the subsequent steps of the algorithm.

- *Splitting Clusters*
  If a set of tracks exceed the removal distance but among each other they are still close enough to stay together, the cluster is split up. A cluster can split up into any amount of new clusters.

Track Association

All tracks which are not associated with any cluster, including the tracks which were removed from clusters in the re-clustering, are compared to the set of all clusters. If a track is close enough to other tracks in any other existing cluster the track will be added to this. If a track fits into several clusters these may be merged together depending on the parameterization of the algorithm, where merging clusters should be done with caution. In addition to a shared track the clusters should have further similarities. Such a similarity measure could be, among others, the yaw angle or the difference of the averaged velocity between two time steps (and so a measure of acceleration).

4.4.3 Hypotheses Generation

After tracking the raw data and clustering the resulting tracks, the output hypotheses $\begin{bmatrix}\mathcal{H}^1 & \mathcal{H}^2 & \cdots & \mathcal{H}^n\end{bmatrix}$ as defined in Equation (4.1) have to be generated. Thus, the task of the hypotheses generation, as depicted in Figure 4.15, is to determine the yaw angle, the best reference point, the position, and an estimation of the dimension of the cluster. Details on how to determine these properties and on reference points

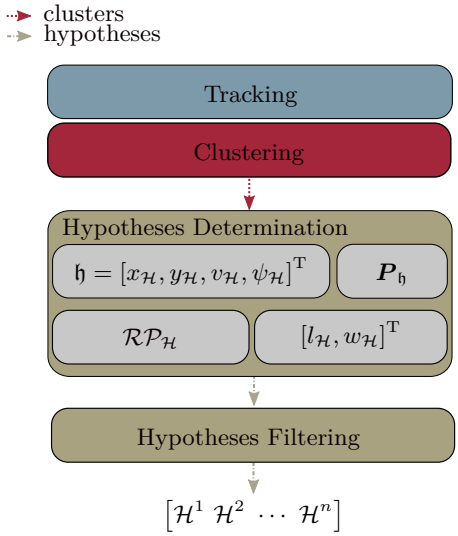


Figure 4.15: Flow chart of the hypotheses generation. Input are the tracks from Figure 4.7.

are pointed out in the following sections. Further, the process of hypotheses filtering, the last step in Figure 4.15, is explained.

Position and Reference Point

Determining the position of the hypothesis vector $\mathfrak{h}$, as defined in Equation (4.1), from a cluster is more than calculating the arithmetic mean of all cluster points. The mean value represents the centroid of the cluster, but the centroid is not always the best solution. This is because the sensor does not deliver measurements from all over the object, but only from one to two sides. This results in the typical I and L shapes as shown in Figure 4.16. The need for defining a point on the object where the given position is valid was already addressed in [Käm07]. This idea

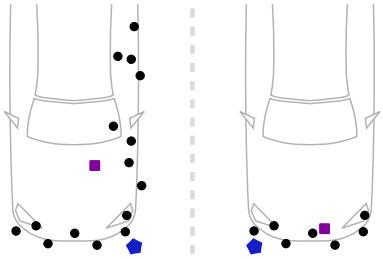


Figure 4.16: Calculating the position as centroid for a L-shape on the left and an I-shape on the right. The raw data tracks are depicted as black circles and the resulting centroid as purple rectangle. The example makes clear, that the centroids are not a sufficient solution for a valid position on an extended object. A position like the blue star would be of higher value if a reference is given where on the extended the position is valid. This reference is called a reference point.

was extended in [SWBH12], where nine distinctive points on extended objects are defined as depicted in Figure 4.17. In this work, these points are called reference

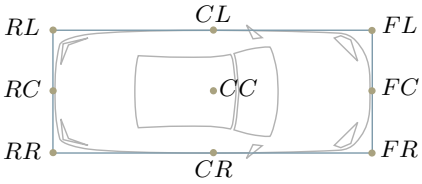


Figure 4.17: Available reference points of an object. The first letter defines the longitudinal position {Front, Center, Rear} and the second letter defines the lateral position {Left, Center, Right}.

point $\mathcal{RP}_{\mathcal{H}} \in \{FL, FC, FR, CL, CC, CR, RL, RC, RR\}$. Fusing information about extended objects from different sources gets more feasible when having reference points. One example: a sensor fusion system with multiple sensors observes the front of an object with one sensor and the back of it with another sensor. If there are no reference points given by the preprocessing, the association of these two measurements is difficult. Using reference points allows a sophisticated subsequent algorithm to recognize that the two measurements originate from the same object but from different sides under the assumption that a classification of the perceived object is done to know the probable extent of the object. This was done for example in [Mei16], where multiple LiDARs are mounted at a public intersection. At this intersection multiple kinds of road users like pedestrians, cars, and trucks are

perceived and tracked with measurements from multiple sides.
Determining the reference point strongly depends on the yaw angle of the perceived object, since the reference point should always be the point of an object which is observed best. In this work, the reference point is always set to one of the corners since there is no information about the real dimension of the object and therefore the center points $\{FC, CL, CC, CR, RC\}$ can not be determined correctly. The best observable corner in most cases is the one closest to the observation system. A drawback of this approach is, that the corners are not always visible or detected, e.g. in case of a bicycle from behind or a vehicle at high distance. This drawback is partly solved by having yaw angle information from the raw data tracks which allows a relatively good estimate of the bounding box and the dimension. In case of a bicycle this might solve the problem. In case of a vehicle at higher distance and a very small amount of points it can not be guaranteed, that the measurements were caused around the corner. This problem can only be solved over time by the subsequent object tracker.

Yaw Angle and Velocity

One big advantage of the preprocessing presented in this work is a reliable estimate of the yaw angle and the velocity, which is also needed for the reference point and dimension determination. Both values are extracted from the raw data tracks in each cluster. Even after filtering of the data, jittering and noisy measurements cause angle and velocity outliers. A stable output is achieved by calculating the median of the yaw angle and the velocity over all tracks of the cluster.

Dimension

The dimension is obtained in every cycle by calculating a rectangular bounding box surrounding all cluster points. The already determined yaw angle is used as orientation of this box. The dimension is not filtered over time to elude problems with incorrect estimated dimension. The presented preprocessing aims at being a generic algorithm which does not incorporate deeper knowledge about the model of the perceived object. Without more sophisticated models and some assumptions it is not possible to filter the true dimension over time correctly. For that reason, the currently observed dimension of the perceived extended object is used as output. So, a subsequent algorithm can use this information to estimate a correct shape over time by having a reliable yaw angle estimation from the preprocessing.

Covariance

For many subsequent algorithms like object tracking, a further quantity is needed or at least helpful: the covariance. In the here presented approach, the main output is the hypothesis vector $\mathfrak{h}$. It would be a big advantage over simple preprocessing approaches if it is possible to give a reliable covariance $\boldsymbol{P}_{\mathcal{H}}$ for this vector. Given the fact, that there are already tracks t with its corresponding covariances in every cluster, generating a covariance for the cluster of tracks seems not to be a difficult task. Unfortunately, to the knowledge of the author, there are no closed solutions to solve this problem. In this section it is discussed why the standard approaches fail and a possible solution is suggested. This solution was implemented during this work but its quality is not evaluated due to the lack of a reference.

Failing Approaches

Averaging the Available Track Covariances Having already the covariances from the single tracks of a cluster, an average of these covariances could be calculated. These covariances are a measure of uncertainty of the single tracks, but tend to underestimate the uncertainty of the hypothesis. The solid red line in Figure 4.18b represents this approach an makes it obvious, that the covariance of the single tracks and the covariance of the hypothesis are not the same. Nevertheless, these two covariances are not completely uncorrelated, but finding a closed solution for this relation seems not to be realistic.

Sample Variance and Covariance Having multiple tracks t per cluster, it is possible to calculate the sample variances $[\widetilde{\sigma}_{x_{\mathcal{H}}}, \widetilde{\sigma}_{y_{\mathcal{H}}}, \widetilde{\sigma}_{v_{\mathcal{H}}}, \widetilde{\sigma}_{\psi_{\mathcal{H}}}]$ for all elements of $\mathfrak{h}$ with the standard equation for the sample variance of a random variable a:

$$\widetilde{\sigma}_a = \frac{1}{N-1} \sum_{i=1}^{N} (a_i - \overline{a}) \tag{4.19}$$

Further, the covariance $\widetilde{\sigma}_{xy}$ for the position can be obtained using the equation of the sample covariance for two random variables a and b:

$$\widetilde{\sigma}_{ab} = \frac{1}{N-1} \sum_{i=1}^{N} (a_i - \overline{a}) \left(b_i - \overline{b}\right). \tag{4.20}$$

Under the assumption, that the position, the yaw angle, and the velocity are uncorrelated, the covariance matrix is given by:

$$\boldsymbol{P}_{\mathfrak{h}} = \begin{bmatrix} \widetilde{\sigma}_{x_{\mathcal{H}}} & \widetilde{\sigma}_{x_{\mathcal{H}}y_{\mathcal{H}}} & 0 & 0 \\ \widetilde{\sigma}_{x_{\mathcal{H}}y_{\mathcal{H}}} & \widetilde{\sigma}_{y_{\mathcal{H}}} & 0 & 0 \\ 0 & 0 & \widetilde{\sigma}_{v_{\mathcal{H}}} & 0 \\ 0 & 0 & 0 & \widetilde{\sigma}_{\psi_{\mathcal{H}}} \end{bmatrix}. \tag{4.21}$$

As already pointed out in the calculation of the position $[x_{\mathcal{H}}, y_{\mathcal{H}}]^{\mathrm{T}}$ and depicted in Figure 4.16, a mean value is already erroneous. Calculating the sample variance and covariance for an L or I-shape would be directly connected to the observed size of the extended object. An object which is far away may cause only a few measurement points close to each other. The (co)variance for this would be rather small. At the same time, a close object causes many measurement points spread over the whole object. The (co)variance for the close object would be much bigger than for the far object. Further, using the sample covariance with the mean values $\overline{\boldsymbol{x}}$ and $\overline{\boldsymbol{y}}$ would be incorrect anyway, since the position of the object is given for the reference point $\mathcal{RP}_{\mathcal{H}}$. Using the given reference point in Equation (4.20) instead of the mean value even increases this problem as depicted with the dashed orange line in Figure 4.18b.

Proposed Solution

Uncorrelated Covariance Along the Shape This approach is the proposed solution of this work, where the main requirements to the covariance are: it is valid at the position of the reference point $\mathcal{RP}_{\mathcal{H}}$ and it does not underestimate the covariance. This definition is an information which subsequent algorithms can use without being distracted by too small uncertainties. Using the reference point as basis the covariance can't be more than an upper bound, since the reference point is the corner of the surrounding bounding box. Thus, all tracks t are inside of this box and are not symmetrically distributed around the basis $\mathcal{RP}_{\mathcal{H}}$. The covariance for the position $[x_{\mathcal{H}}, y_{\mathcal{H}}]^{\mathrm{T}}$ is calculated as the sample covariance from Equation (4.20) over all track positions $[x^t, y^t]^{\mathrm{T}}$ for $t = 1, \ldots, N$, where the mean values are replaced by the position of the reference point:

$$\widetilde{\boldsymbol{P}}^{0}_{xy} = \frac{1}{N-1} \sum_{t=1}^{N} \left(x^t - x_{\mathcal{H}}\right) \left(y^t - y_{\mathcal{H}}\right). \tag{4.22}$$

This equation holds only true for yaw angles equal to zero: $\psi_{\mathcal{H}} = 0$. Otherwise, the uncertainties are clearly overestimated. Therefore, all track positions need to be

shifted by $[x_{\mathcal{H}}, y_{\mathcal{H}}]^{\mathrm{T}}$ and rotated counter clockwise by the angle $\psi_{\mathcal{H}}$:

$$\begin{bmatrix} x^{t,r} \\ y^{t,r} \end{bmatrix} = \boldsymbol{R}(\psi_{\mathcal{H}}) \left(\begin{bmatrix} x^{t} \\ y^{t} \end{bmatrix} - \begin{bmatrix} x_{\mathcal{H}} \\ y_{\mathcal{H}} \end{bmatrix} \right) \tag{4.23}$$

where $\boldsymbol{R}(\varrho)$ is the rotation matrix for angle ϱ and $[x^{t,r}, y^{t,r}]^{\mathrm{T}}$ are the rotated positions. To calculate two uncorrelated variances for the directions in x and y the tracks t are assigned to two sets T_x and T_y:

$$T_x = \{t | x^{t,r} \leq y^{n,r} \forall n \in 1 \dots N\} \tag{4.24}$$

$$T_y = \{t | y^{t,r} < x^{n,r} \forall n \in 1 \dots N\}. \tag{4.25}$$

The position variances are then obtained using the tracks of the corresponding set:

$$\widetilde{\sigma}_{x_{\mathcal{H}}} = \frac{1}{|T_x| - 1} \sum_{t \in T_x} x^{t,r} \tag{4.26}$$

$$\widetilde{\sigma}_{y_{\mathcal{H}}} = \frac{1}{|T_y| - 1} \sum_{t \in T_y} y^{t,r}. \tag{4.27}$$

Thus, the covariance matrix is given with:

$$\widetilde{\boldsymbol{P}}^{r}_{xy} = \begin{bmatrix} \widetilde{\sigma}_{x_{\mathcal{H}}} & 0 \\ 0 & \widetilde{\sigma}_{y_{\mathcal{H}}} \end{bmatrix}. \tag{4.28}$$

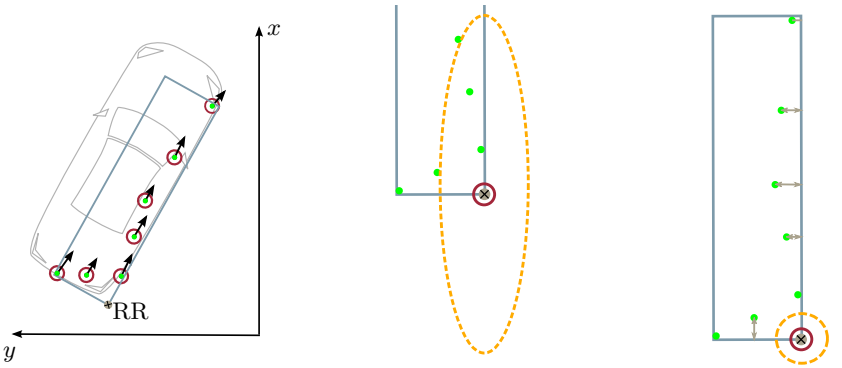
This covariance matrix has to be rotated clockwise by the angle $\psi_{\mathcal{H}}$:

$$\boldsymbol{P}_{\mathfrak{h},xy} = \boldsymbol{R}(-\psi_{\mathcal{H}}) \widetilde{\boldsymbol{P}}^{r}_{xy} \boldsymbol{R}(-\psi_{\mathcal{H}})^{\mathrm{T}}. \tag{4.29}$$

where $\boldsymbol{P}_{\mathfrak{h},xy}$ is the resulting covariance matrix for the position. Furthermore, the variance of the angle $\psi_{\mathcal{H}}$ and the velocity $v_{\mathcal{H}}$ is calculated using the sample variance from Equation (4.19). Assuming that the position, the angle, and the velocity are uncorrelated, the hypothesis covariance matrix is given with:

$$\boldsymbol{P}_{\mathfrak{h}} = \begin{bmatrix} \boldsymbol{P}_{\mathfrak{h},xy} & \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix} \\ \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix} & \begin{matrix} \widetilde{\sigma}_{v_{\mathcal{H}}} & 0 \\ 0 & \widetilde{\sigma}_{\psi_{\mathcal{H}}} \end{matrix} \end{bmatrix} \tag{4.30}$$

This covariance is depicted in Figure 4.18c as dashed orange circle.



(a) Example of a perceived object with the tracked raw data points and the resulting bounding box. The solid red circles represent the uncertainty ellipse of the track covariance.

(b) Estimated covariance of the position using the sample covariance on all points in orange (dashed). For comparison the average uncertainty of the single tracks in red (solid).

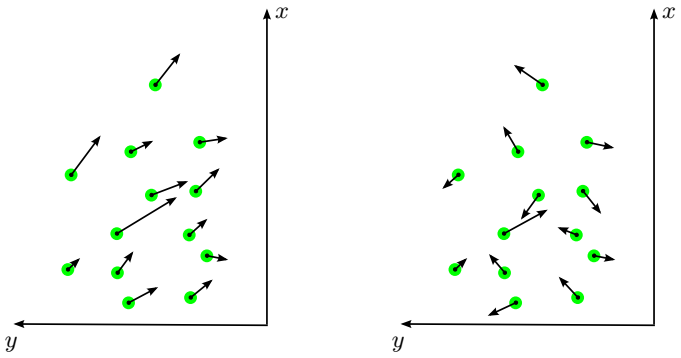
(c) Proposed solution: Tracks are associated to either one of the directions to calculate uncorrelated variances. The resulting uncertainty ellipse is depicted as dashed orange line. In solid red the average uncertainty of the single tracks.

Figure 4.18: Estimating an upper bound of the position variances. The tracked raw data points are depicted in green and the uncertainty of the single tracks in red (solid). The black arrows represent the velocity and the angle information of the tracks. The resulting bounding box is shown in blue.

Hypotheses Filtering

The last step of the hypotheses generation as shown in Figure 4.15 is the hypotheses filtering. It is an optional step of the algorithm which could also be done by a subsequent algorithm to adapt this step to the respective needs. In algorithms like vehicle tracking it is very common to use background knowledge, e.g. a dimension minimum, to filter incoming data. In this section, an approach to filtering data is presented without the need for knowledge about the type of object of interest. The main assumption is, that an object of interest is moving. With this, it follows that all raw data tracks belonging to the cluster of an hypothesis should behave in a similar manner. This characteristic was already used in the clustering step, but it is still possible, that tracks are clustered together even if their behavior is not exactly

the same, where the behavior is defined by their velocity and yaw angle. A very good feature for filtering purposes is the variance of the yaw angle in a cluster. Discarding objects with high variances of the yaw angle reduces the amount of false positives significantly. Such high variances inherit from clusters without one preferred moving direction. The example in Figure 4.19 illustrates the same cluster twice: with a low and a high variance of the yaw angle. With such a filtering non-moving objects



(a) Cluster with a low variance of the yaw angle.

(b) Cluster with a high variance of the yaw angle.

Figure 4.19: Discarding a cluster using the variance of the yaw angle. The green dots represent the track positions and the black arrows the yaw angle where the length of the arrows illustrates the velocity. The same artificial cluster is depicted twice. In a) the variance of the angle is rather low and in b) it is quite high. A low variance represents the case, that all tracks of the cluster move approximately in the same direction. A high variance represents the case, that the cluster inherits from a non-moving object.

tend to be discarded. Table 4.3 shows a comparison of the results using different thresholding filters with the angle variance $\mathrm{Var}_{OA} = \sigma_{OA}^2$ and its standard deviation σ_{OA}. The filtering values are obtained by brute force optimization running the following evaluation in Section 4.5. For $\sigma_{OA} = \sqrt{3}^\circ$ the filtering is very cautious which does not remove relevant hypotheses, called true positive (TP), and is used in the upcoming evaluation. A standard deviation of $\sigma_{OA} = 1^\circ$ already reduces the FPs to nearly 63%. A more strict threshold of $\sigma_{OA} = \sqrt{0.5}^\circ$ reduces the FPs even more, but the TP rate starts to decrease. Most of the FPs inherit from stationary objects. A minimum velocity can reduce the false positive rate again to 45%. This equals a reduction to 29% in total.

	scenario Nr. i, roundabout			
σ_{OA}^2 in $[^\circ]^2$	3	1	0.5	1
v_{min} in $\frac{m}{s}$	0	0	0	0.5
false positives	9305	5916	5277	2687

Table 4.3: The amount of FPs can be reduced using the yaw angle standard deviation of the angle σ_{OA} or a minimum velocity v_{min} of the cluster. For $\sigma_{OA} = 1^\circ$ the amount of false positives can be reduced clearly. For $\sigma_{OA} = \sqrt{0.5}^\circ$ the reduction is even better, but the amount of TPs also reduces. Thus, the performance starts to decrease . (For reference: the chosen scenario is Nr. i in Table 4.4 and the used method is GCEM)

4.5 Evaluation

In this section the presented approach for preprocessing data from high density distance measuring sensors is evaluated. The evaluation is done using various real-data scenarios with the sensor setup introduced in Section 4.3. A summary of all evaluated scenarios is given in Table 4.4. In this work it is assumed that the raw data preprocessing is mainly used to detect moving objects. Thus, a generated hypothesis for a static object is regarded as a false positive (FP). To reduce the amount of FPs the hypotheses are filtered, as explained in Section 4.4.3, with a very cautious threshold of $\sigma_{OA} = \sqrt{3}^\circ$. The ground truth data for the evaluation of the vehicle perception was generated by using two GeneSys automotive dynamic motion analyzers (ADMAs). An ADMA provides high precision information about the state of the ego vehicle using a global navigation satellite system (GNSS) and an inertial measurement unit (IMU). One ADMA in the observing vehicle and one in the observed one allows a very accurate differential positioning. In case of perfect GNSS receiving conditions and available correction data for the differential GNSS, the ADMA allows a global position accuracy of up to $0.02m$ [Gen09]. Using two ADMAs in different vehicles, the uncertainty increase accordingly. Nevertheless, this is almost one order of magnitude better than the sensor accuracy given in Table 4.1. In practice, one has to deal with various inaccuracies, since the receiving conditions are not perfect all the time. Further, the temporal alignment of the perception system to the reference system is very important. A deviation of only a few milliseconds can cause position errors up to meters for higher velocities. Moreover, the quality of the single measurements points of the used LiDAR sensors strongly depends on the surface they are reflected by, e.g. it is noticeable, that pure black vehicles often do not cause any measurement points. Further, highly reflecting surfaces, like number plates, may cause additional inaccuracies. To evaluate the pedestrian and bicycle perception the sensor raw data was used to generate the ground truth data by hand.

Nr.	Scenario	Figure	Brief Description
i	standstill roundabout	4.20	The ego object is standing at a rather flat roundabout. The perceived object drives a bit more than two rounds in the roundabout.
ii	standstill receding	4.23	The ego object is standing still, the perceived object recedes on a straight rural road in a narrow serpentine line.
iii	standstill approaching	4.26	The ego object is standing still, the perceived object approaches on a straight rural road in a wide serpentine line.
iv	standstill crossing	4.29	The ego object is standing diagonally at the side of a rural road. The perceived object crosses on the road.
v	following	4.32	The ego object follows the perceived object on a rural road.
vi	following while catching up	4.35	The ego object follows the perceived object while catching up on a rural road.
vii	following while receding	4.38	The ego object follows the perceived object while this is receding on a rural road.
viii	Pedestrian & Bicycle	4.42	The ego object is standing still. A pedestrian and a bicycle moves into the field of view after each other (not at the same time).

Table 4.4: Summary of the evaluated scenarios.

This manual labeling process can not be assumed to be perfect, since it depends on the sensor measurements, the calibration of the sensors, the human influence, and the difficulties in defining exact rules for the labeling process. Comparing the algorithm results against such an inaccurate ground truth can only give an estimate of the algorithm quality. Alltogether, four different algorithm versions are evaluated as explained in Table 4.5. The difference of these four versions is the distance measure (Section 4.4.2) and the usage of the cluster memory from the GC. The underlying tracking of the raw data is always the same. These four methods are compared to the box fitting algorithm presented in Section 4.1 as a representative of state of the art algorithms. In the pedestrian and bicycle scenario (Nr. viii in Table 4.4) the BoxFit can not be applied, since this algorithm is optimized to detect vehicles. The search

Method	Shortcut	Description
Euclidean	EUC	The clustering is done with a position based DBSCAN algorithm [EKSX96] using the Euclidean distance. There is no knowledge about the history used in the cluster process.
Bhattacharyya	BHAT	The clustering is done using a DBSCAN without incorporating the history. As distance measure, the Bhattacharyya distance (BD) [Bha46] in Equation (4.11) is used.
Graph Clustering Bhattacharryya	GCB	The difference to the BHAT method above is the additional use of the cluster history used by the graph clustering (GC) [SS06].
Graph Clustering Euclidean Mix	GCEM	As in the GCB method the graph clustering (GC) is used additionally. As distance, a mixture of the Euclidean distance, the yaw angle difference, and the velocity difference as presented in Section 4.4.2 is used.

Table 4.5: Evaluated algorithm versions.

for specific I and L shapes does not work for circular objects like pedestrians and has an only limited ability to detect bicycles. Thus, the box fitting is not available for comparison in the scenario Nr. viii. The evaluated quantities are described in Table 4.6. The evaluation of every sequence is a compound of three parts:

- first part is the trajectory of the perceived object in vehicle coordinates. If the perception vehicle and the perceived object are moving at the same time, it is very difficult to understand the scenario if it is given in vehicle coordinates and the movement of the ego object is not visible. In this case, the trajectories are given in a common Cartesian coordinate system using the universal transverse mercator (UTM) coordinate system as basis. The time is always color coded from cold to warm colors.

- second part is a table with the evaluation quantities given by Table 4.6. The values are color coded column wise from white to black, where white is the best and black the worst result. During the evaluation it was observed, that the presented approaches outperform the box fitting algorithm by far. So, the box fitting is excluded from the color coding. Otherwise, the differences between the other four methods are not visible. The error of the velocity is not evaluated for the box fitting since it is not able to estimate it. The evaluation

Quantity	Description
pos. err. in m	The root mean square error (RMSE) of the position. Evaluated is the closest distance of the reference points of the four corners to reduce the impact of the dimension estimation.
ori. err. in °	The RMSE of the yaw angle.
false positives	The amount of false positive measurements without the use of any reduction method.
drop in %	The drop evaluates the portion of time stamps where the cluster method does not generate a measurement. It is in relation to the amount of time stamps between the moment of the first and the last measurement any of the methods generates. A drop of 0% would be a perfect result.
max. dist. in m	The maximum measured distance to the object.
vel. err. in $\frac{m}{s}$	The RMSE of the velocity.

Table 4.6: Evaluation quantities.

always starts with the first detection of any algorithm and ends with the last detection or the end of the scenario.

- third part is the error evaluation over time in two plots: one of the position error and on of the yaw angle error. In the same plots, the distance of the perceived object, only if detected, is depicted with a gray dash dotted line. The gray boxes represent times, where the method was *not* able to generate a measurement.

4.5.1 Evaluation of the Vehicle Perception

In this section the vehicle perception capabilities of the presented preprocessing algorithms are evaluated. The evaluation is given in detail for the scenarios Nr. i to Nr. vii in Table 4.4.

Scenario Nr. i, Standstill Roundabout

In scenario Nr. i in Table 4.4, the perceived object drives a bit more than two rounds in a roundabout. The ego object is standing still. Figure 4.20 shows the

reference trajectory. The increasing height of the trajectory is not real and was added to allow a better overview. The object leaves the roundabout at the exit the ego object is entering. Table 4.7 shows the evaluation quantities for the roundabout

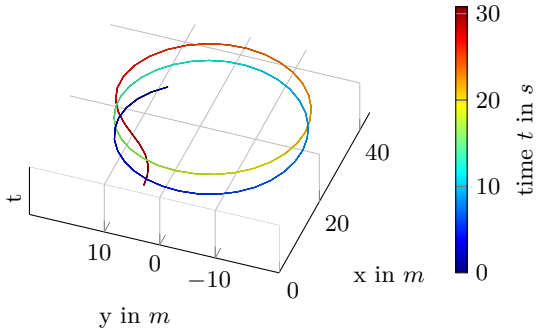


Figure 4.20: Reference trajectory of the perceived object in the scenario (Nr. i in Table 4.4). For the reason of a better overview, the height increases over time.

scenario. The position and yaw angle error of the four evaluated methods presented

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.64	8.21	6042	15.52	47.69	3.16
BHAT	0.74	7.99	10436	12.21	47.98	3.11
GCB	0.73	8.13	14715	11.45	47.96	3.14
GCEM	0.73	8.20	9305	10.69	47.98	3.16
BoxFit	1.16	47.99	5977	58.78	46.04	

Table 4.7: Evaluated quantities of the roundabout scenario (Nr. i in Table 4.4) corresponding to Table 4.6. The maximum reached distance can not get higher than approximately $48m$ since the roundabout is not bigger than that. Beside the quite high angle error of the BoxFit the very high drop rate catches one's eye. This is caused by partial occlusions on the opposite side of the roundabout the BoxFit is not able to deal with.

is approximately the same. They differ most in the amount of FPs and the drop rate. Regarding position and yaw angle, the BoxFit is clearly outperformed. The quite low amount of FPs of the BoxFit is due to the highly specific adaption to vehicles and a very high measurement drop rate. This adaption allows to reject a lot of smaller objects. The drawback is, that these smaller objects might be pedestrians, bicycles, motorbikes, etc. How the amount of FPs of the other approaches can be further reduced is addressed in the Subsection *Hypotheses Filtering* of Section 4.4.3. The EUC seems to perform a bit better in this scenario than the others, but this is at the cost of the drop rate. The error over time in Figures 4.21 and 4.22 shows that the BoxFit is acceptable when the reference vehicle is crossing the ego vehicle and then receding in the roundabout. After passing the turning point the vehicle is lost quite fast. The other methods are able to generate measurements nearly all the time because of the underlying Kalman filtering.

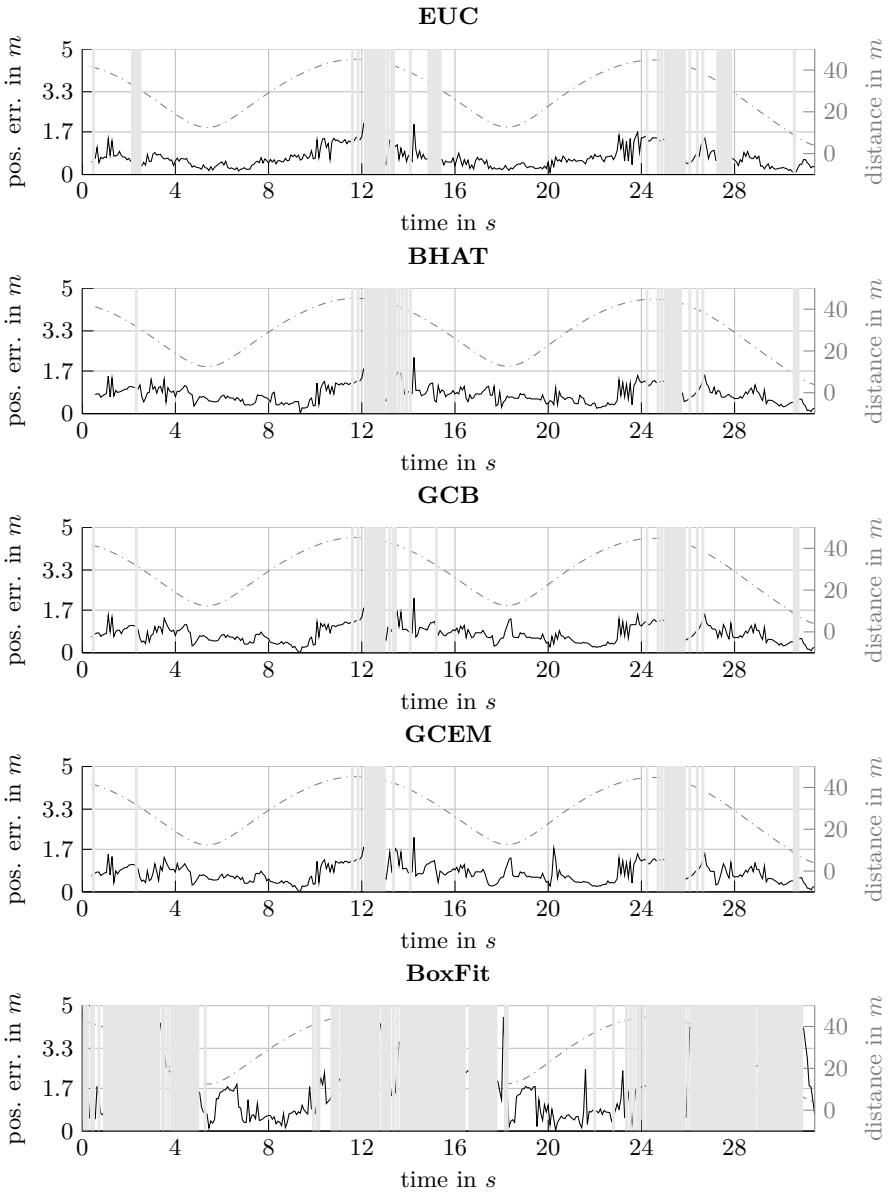


Figure 4.21: Position error over time for scenario Nr. i in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

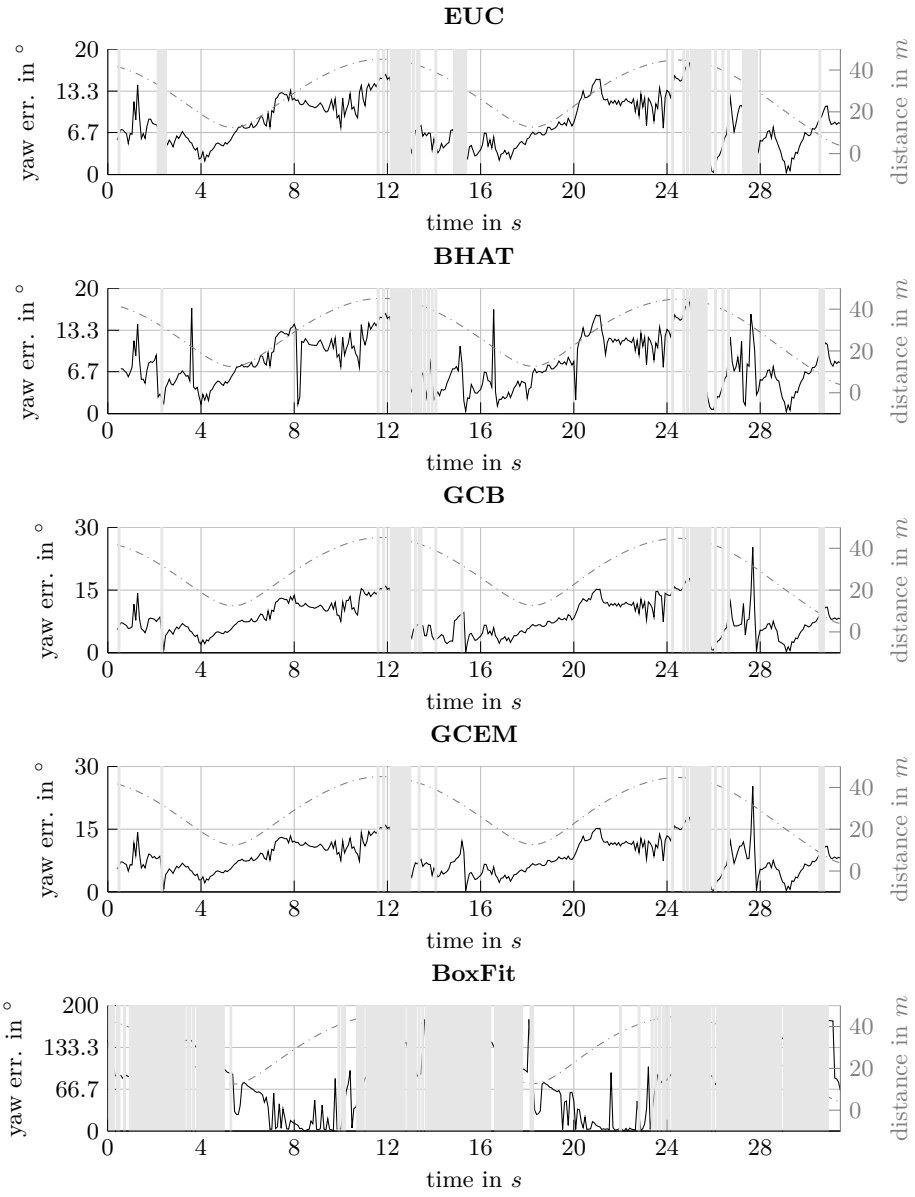


Figure 4.22: Angle error over time for scenario Nr. i in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Scenario Nr. ii, Standstill Receding

In this scenario, the ego vehicle is standing still on a rural road. The perceived object is receding in a narrow serpentine line as depicted in Figure 4.23. This movement was

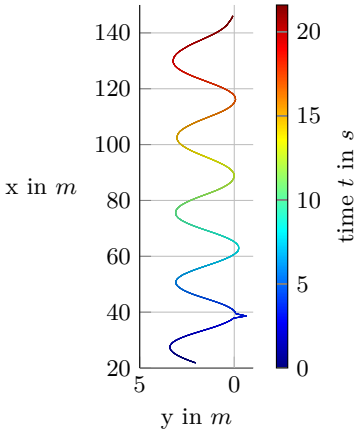


Figure 4.23: Reference trajectory of the perceived object in the scenario Nr. ii in Table 4.4. The perceived objects recedes in a narrow serpentine line.

chosen to evaluate the dynamic behavior of the presented approach. The evaluated quantities in Table 4.8 show, that this is a scenario the BoxFit handles quite good. The position error of the BoxFit is lower than of every other method. Regarding the yaw angle error this is the only scenario the fitting approach works very well and the new approaches are rather poor. Only when taking a look at the measurement drop rate the advantage of the new approaches reveal their advantage, since the BoxFit has a drop rate far beyond 50%. The explanation for this behavior can be found in the detailed plots in Figures 4.24 and 4.25. The difference in the drop rate is quite easy to see: the BoxFit is not able to handle this scenario in more than approximately $50m - 60m$ of distance. In less distance the position error is very good. The problem of the other methods, like the GCEM, in this scenario are the fast changes of the yaw angle. A vehicle driving a normal curve behaves quite smooth and does not change the yaw angle very frequently. The raw data tracking is optimized to handle such a behavior. In case of a narrow serpentine line the yaw angle changes are too fast. Figure 4.24 shows that the algorithm expects the vehicle to drive straight. The error increases and decreases with the frequency of the driven curves. The position error suffers from this too slow yaw angle adaption since the reference point is calculated using the yaw angle. Driving the serpentine line in wider curves like in scenario Nr. iii, this problem is hardly visible.

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.35	12.96	529	23.81	144.07	2.53
BHAT	0.35	12.97	747	25.64	143.99	2.53
GCB	0.36	12.64	843	17.95	124.38	2.44
GCEM	0.40	12.64	749	12.09	145.29	2.45
BoxFit	0.20	2.23	174	71.80	61.58	

Table 4.8: Evaluated quantities of the scenario Nr. ii in Table 4.4. The BoxFit can handle the narrow serpentine line better than the other approaches. Only the measurement drop rate unveils the disadvantage of the fitting algorithm.

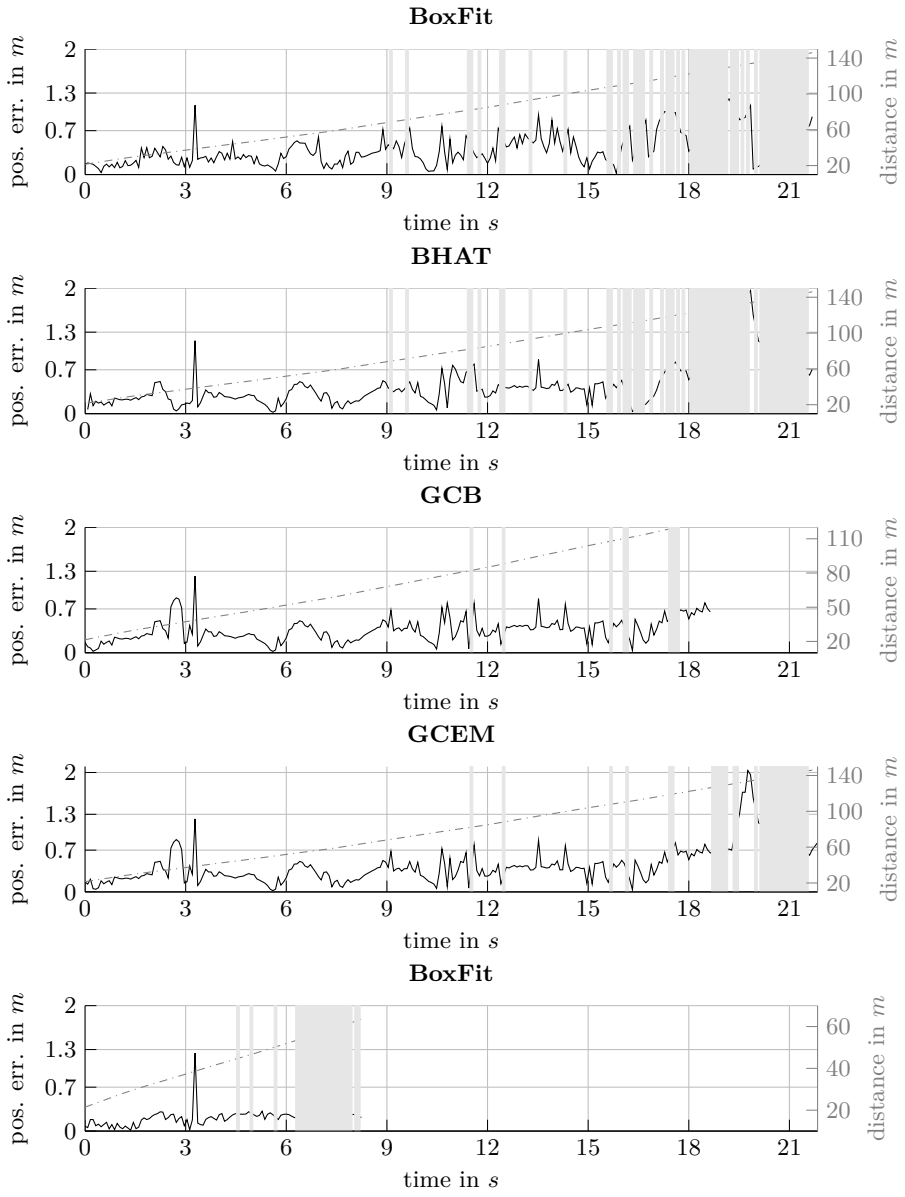


Figure 4.24: Position error over time for scenario Nr. ii in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

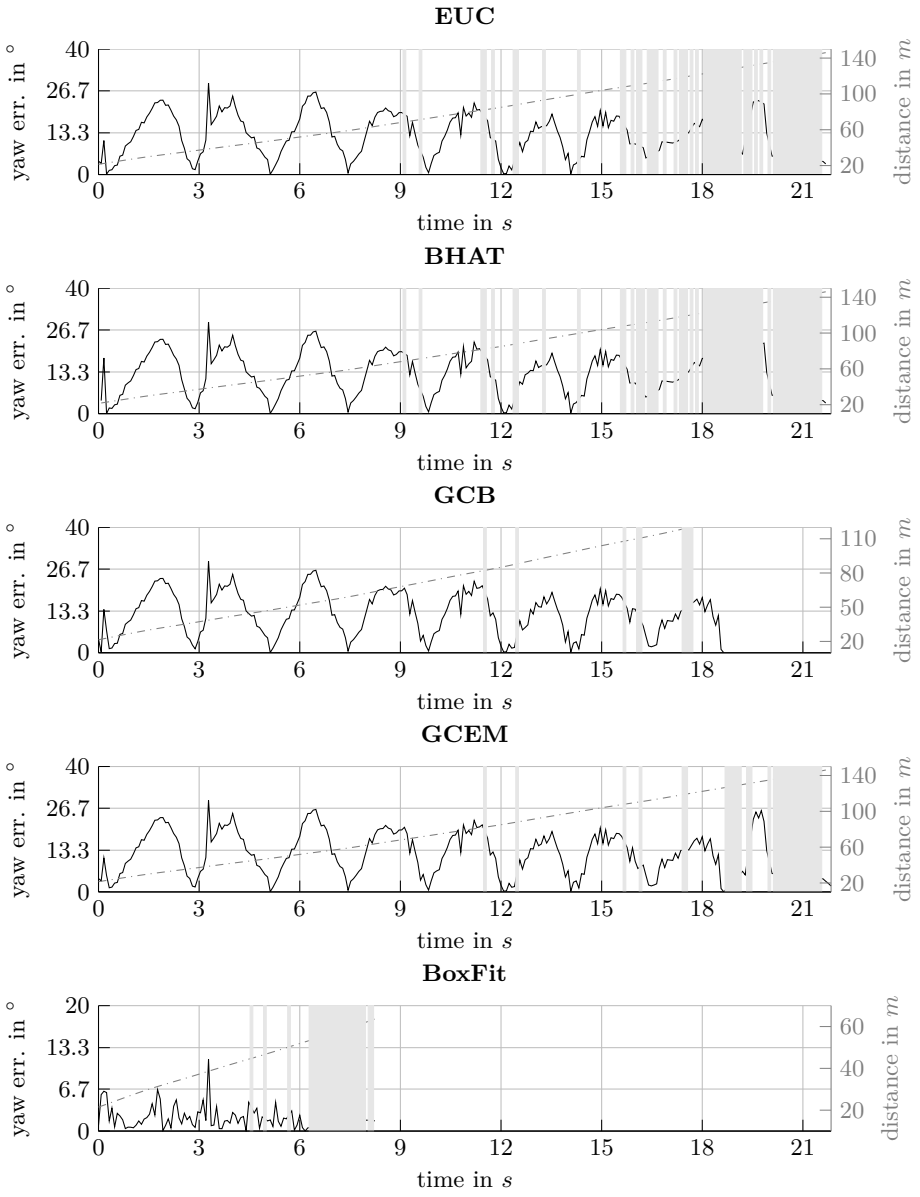


Figure 4.25: Angle error over time for scenario Nr. ii in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Scenario Nr. iii, Standstill Approaching

In this scenario the ego vehicle is standing still on a rural road. The perceived object approaches in a serpentine line but not as narrow as in scenario Nr. ii. The trajectory is depicted in Figure 4.26. The quantities in Table 4.9 show another major

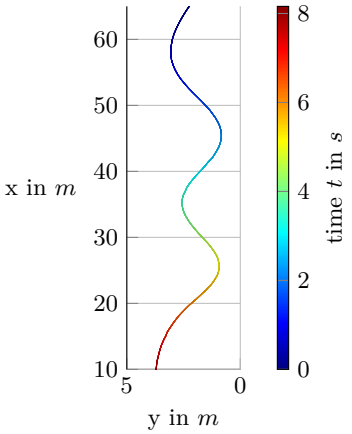


Figure 4.26: Reference trajectory of the perceived object in the scenario Nr. iii in Table 4.4. The perceived objects approaches in a serpentine line.

drawback of any fitting approach which does not incorporate any history: the yaw angle estimation can not decide whether an object is approaching or receding. In this work the used BoxFit was designed to perceive cars from behind. Thus, the yaw angle estimation has an error of over 160° since the perceived object is approaching. In this case, a perfect result of the BoxFit would be 180°. Taking a look at the other quantities shows, that the GCEM method again would be a good compromise because of the small drop rate. The details of the yaw angle error in Figure 4.28 shows that the presented approaches might be a bit too slow for this scenario since the error increases in the parts of the scenario with a higher dynamic behavior, but in comparison to the narrow serpentine line in scenario Nr. ii the error is very small.

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.36	3.64	296	35.92	63.80	1.57
BHAT	0.29	3.84	422	9.71	63.55	1.58
GCB	0.34	3.95	167	8.74	63.55	1.60
GCEM	0.33	3.72	472	5.83	63.55	1.60
BoxFit	2.46	166.61	100	83.50	28.00	

Table 4.9: Evaluated quantities of the scenario Nr. iii in Table 4.4.

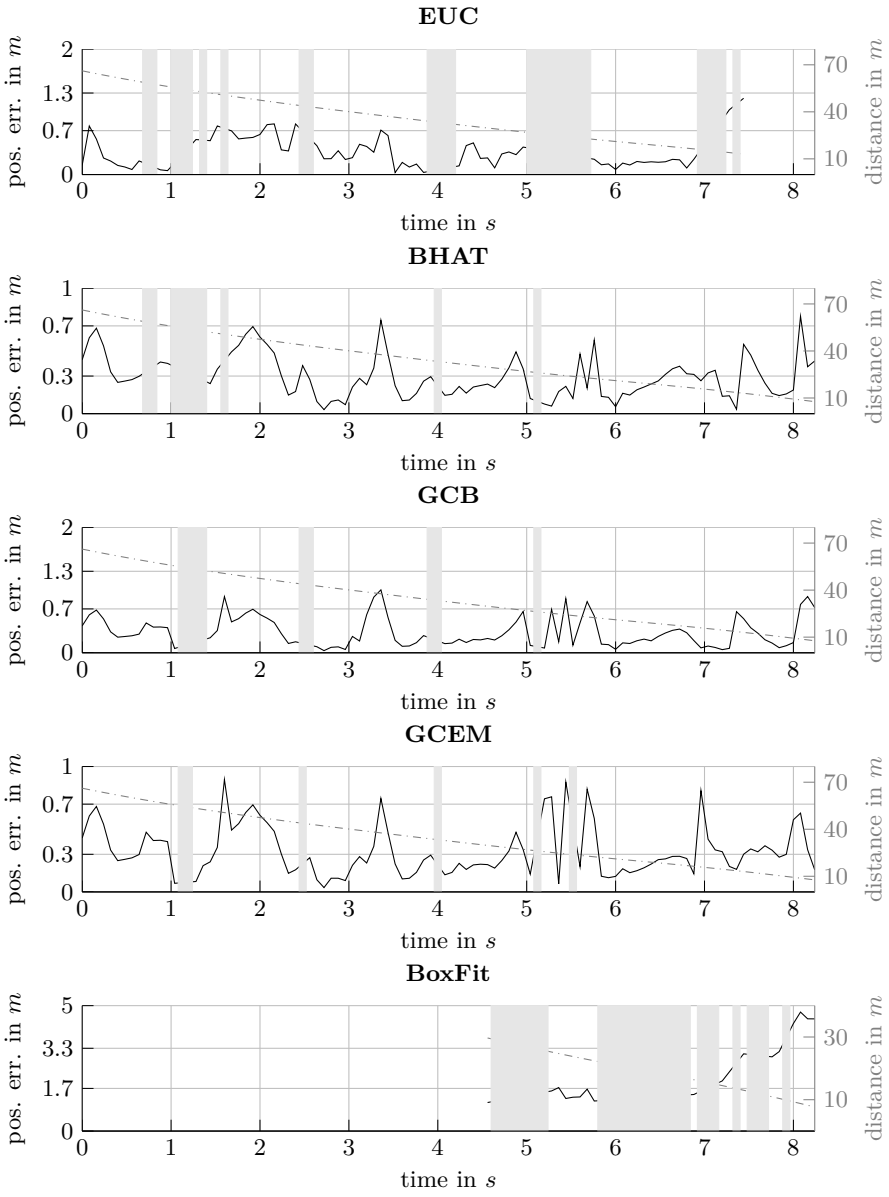


Figure 4.27: Position error over time for scenario Nr. iii in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

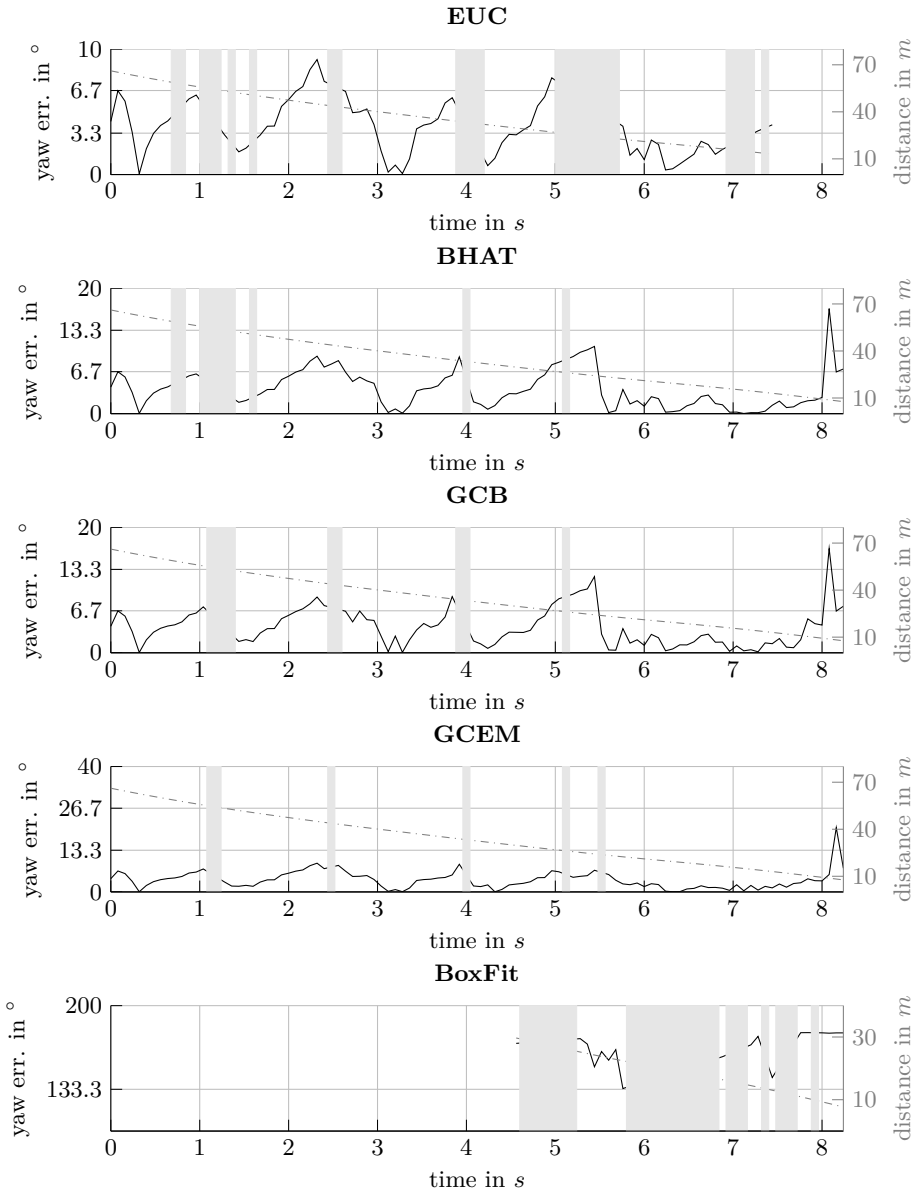


Figure 4.28: Angle error over time for scenario Nr. iii in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Scenario Nr. iv, Standstill Crossing

In this scenario the ego vehicle is standing still at a crossing on a rural road. It is aligned diagonally to the rural road in order to drive onto it. The ground truth data begins in the moment the perceived object enters the field of view of the sensors. This moment is obtained by checking the time of the first measured point of the sensor. Thus, there might be a few milliseconds between entering the theoretic field of view and perceiving the first measurement because of the sensor cycle time of $80ms$. The trajectory of the perceived object is depicted in Figure 4.29 Taking a

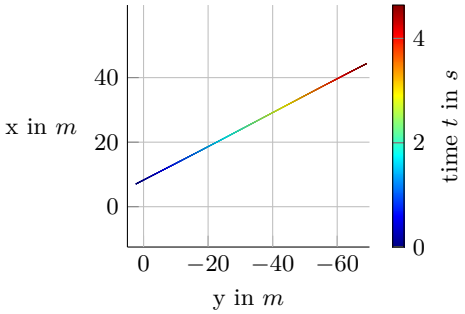


Figure 4.29: Reference trajectory of the perceived object in the scenario Nr. iv in Table 4.4.

short look at Table 4.10 shows, that the presented approaches perform well and all of them are very similar. Once more, the box fitting algorithm has some deficits in

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.32	1.11	85	6.35	42.55	1.30
BHAT	0.43	1.21	130	7.94	42.65	1.27
GCB	0.41	1.08	135	4.76	43.79	1.29
GCEM	0.39	1.10	117	4.76	43.16	1.30
BoxFit	0.60	18.79	106	28.57	41.20	

Table 4.10: Evaluated quantities of the scenario Nr. iv in Table 4.4.

estimating the yaw angle and in a stable perception of the vehicle. In the plot of the yaw angle error in Figure 4.31 one can see, that the tracking approaches need some time steps to correctly initialize the object and after about $1.5s$ the yaw angle error shrinks. Since a crossing vehicle is a very dynamic object, this behavior is expected. Altogether the evaluation in Figure 4.31 shows the problems of the BoxFit in the initialization phase and in distances of more than $50m$.

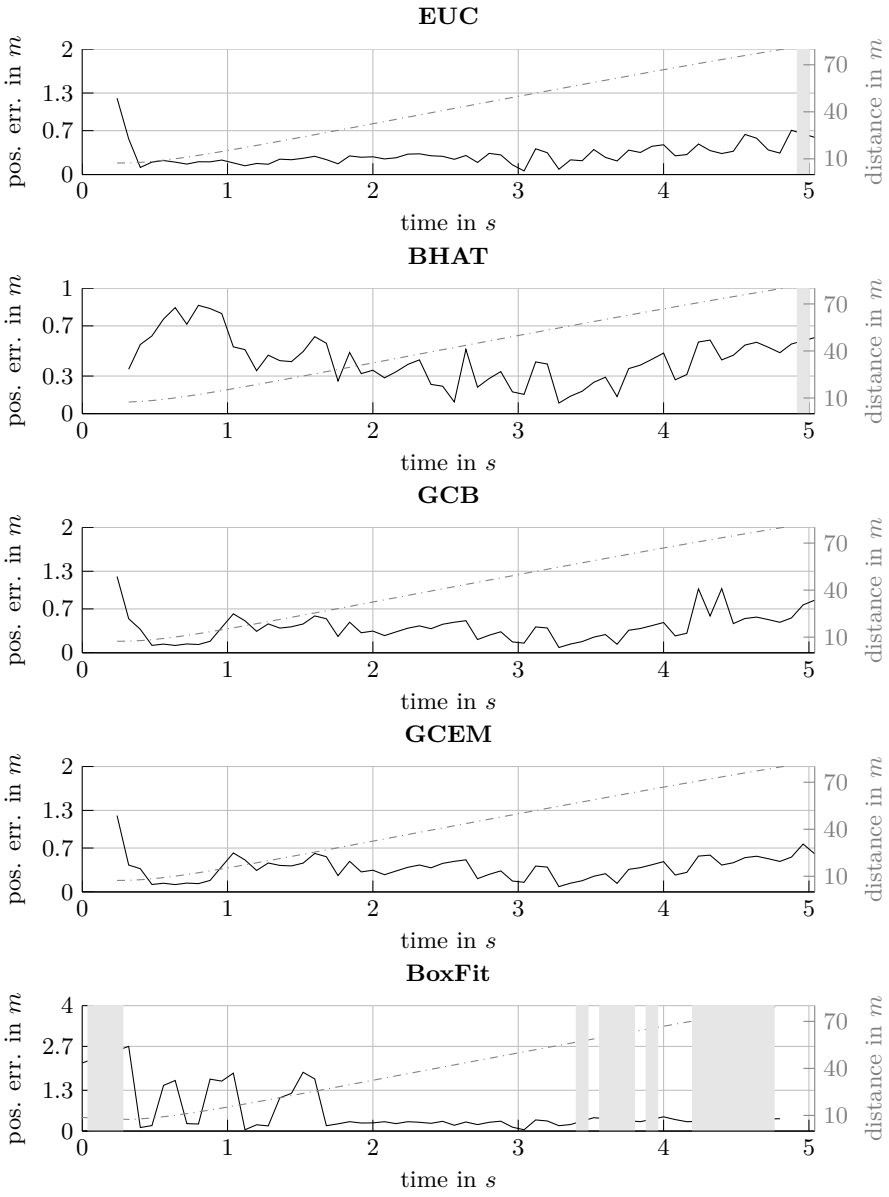


Figure 4.30: Position error over time for scenario Nr. iv in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

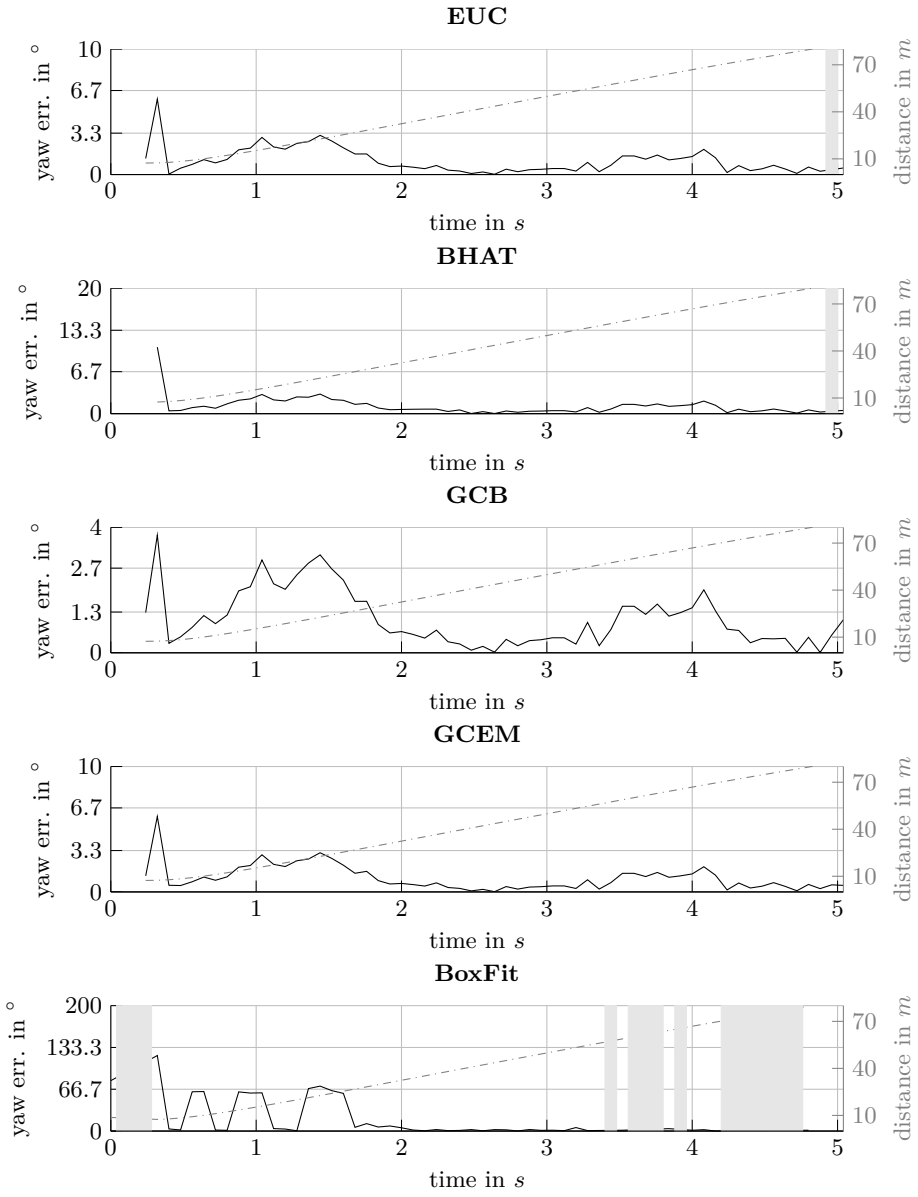


Figure 4.31: Angle error over time for scenario Nr. iv in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Scenario Nr. v, Following

In scenario Nr. v the ego vehicle follows the perceived object on a rural road. The trajectories of the perceived and the ego object in Cartesian coordinates are shown in Figure 4.32. The distance covered during the scenario is approximately 900 meters and lasts about 40 seconds. At the beginning the distance of both vehicles is more

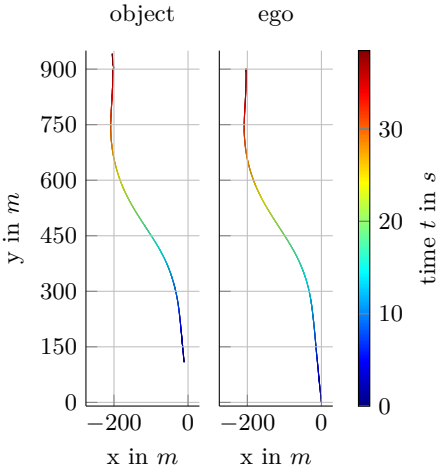


Figure 4.32: Reference and ego trajectory in Cartesian coordinates in Scenario Nr. v in Table 4.4.

than $100m$ and gets less over time. The results of the evaluation of this scenario in Table 4.11 show that this is a scenario, the BoxFit was optimized for. The position and the yaw angle error seem to be quite low, better than of all other methods. Even the maximum distance is comparable to all others. Only the drop reveals the limitations of the BoxFit. Due to its optimization for I and L shapes of vehicles, the BoxFit has, as always, the lowest amount of FPs. Figures 4.33 and 4.34 show the limitations of the BoxFit in detail. Despite the maximum perception distance of over $100m$, it is obvious, that a stable perception is limited to approximately $80m$. The reason for this weak performance in higher distance is that there are only a small amount of points on the perceived vehicle. This leads to a failing fit or rejection because of the size of the object. In comparison to that, the GCEM loses the object only in two time steps during $36s$. This is due to the fact, that the raw data tracking does not need any certain shape. The filtering over time allows the perception with only a few sensor measurements.

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.34	2.10	1686	1.03	109.47	1.06
BHAT	0.35	2.10	3021	1.03	109.42	1.07
GCB	0.37	2.09	3399	0.41	109.42	1.06
GCEM	0.37	2.09	2239	0.41	109.42	1.06
BoxFit	0.32	2.00	1313	34.77	107.52	

Table 4.11: Evaluated quantities of the pursuit scenario (Nr. v in Table 4.4). Obviously this is the favorite scenario of the BoxFit. It has a very small yaw angle error and an even smaller position error. Only the drop rate reveals the problems. In higher distances the BoxFit fails. Thus, it makes sense, that the errors are small in comparison to the other methods, since the BoxFit does not even generate a measurement in difficult parts of the scenario.

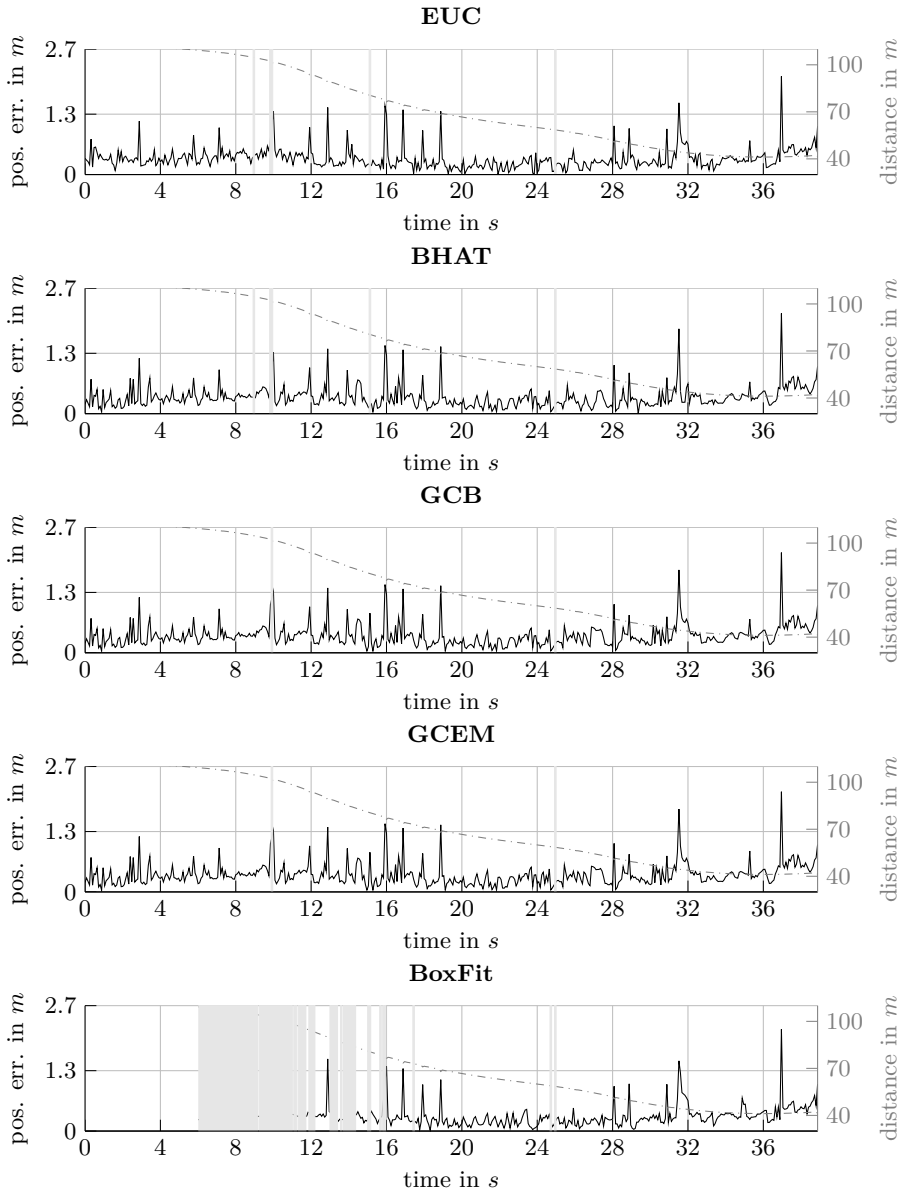


Figure 4.33: Position error over time for scenario Nr. v in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

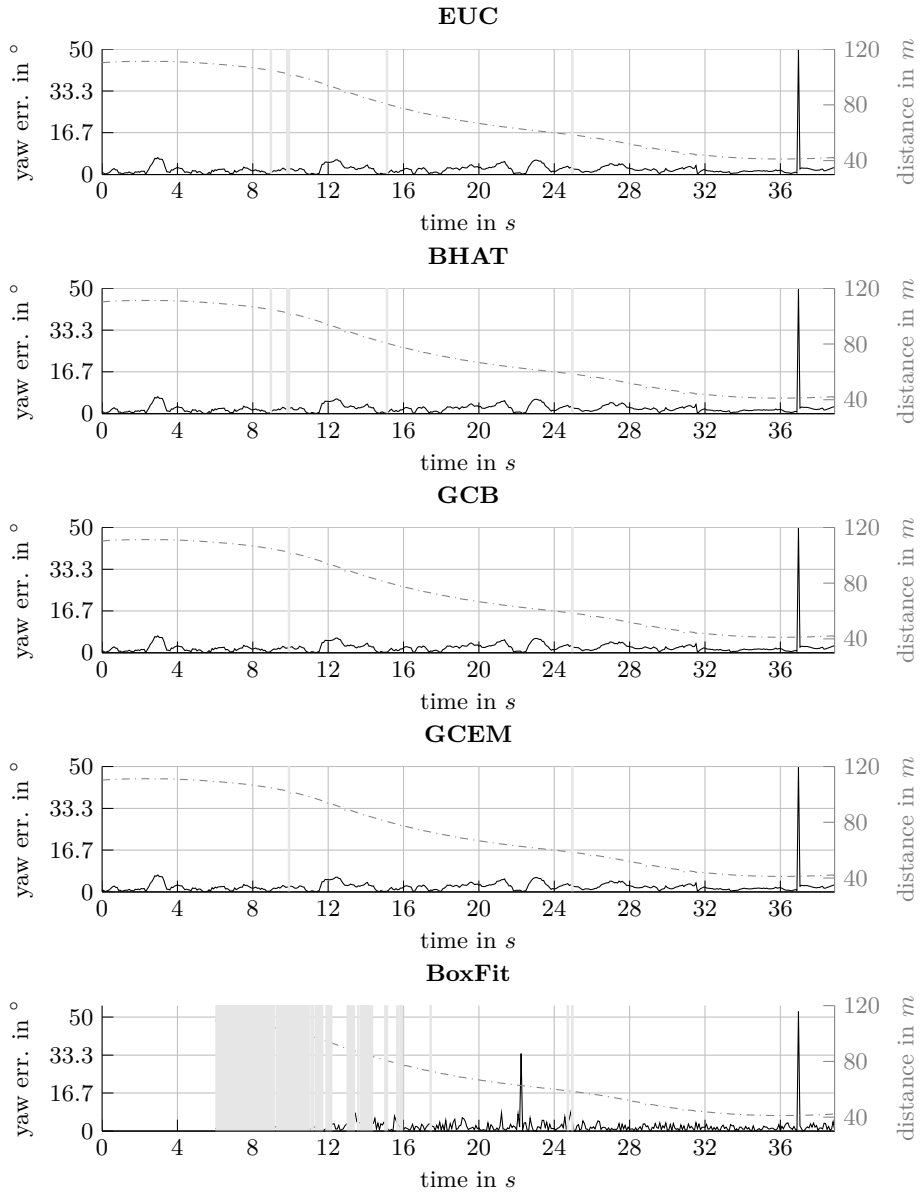


Figure 4.34: Angle error over time for scenario Nr. v in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Scenario Nr. vi, Following while Catching Up

In this scenario the ego vehicle follows the perceived vehicle on a rural road starting in a distance of more than $150m$ and catches up to approximately $20m$. The trajectories of both vehicles are depicted in Figure 4.35. The plots are given in

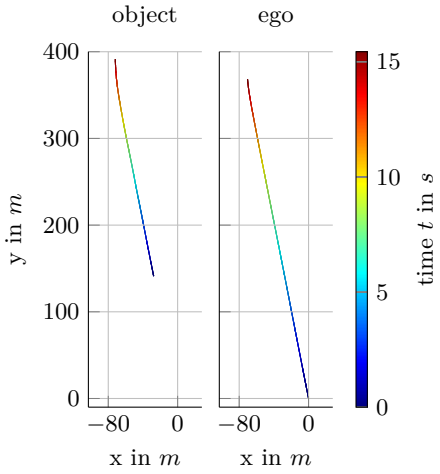


Figure 4.35: Reference and ego trajectory in UTM coordinates in the scenario Nr. vi in Table 4.4.

Cartesian coordinates to visualize the trajectories in relation to each other. The color coded time supports the understanding of the scene. The evaluated quantities are displayed in Table 4.12. This is once more an example of scenarios the BoxFit is optimized for. A low position error and only a small amount of false positives proof that the parameter set of the fitting are well chosen. Even the maximum perceived distance is rather high. The drawback is the measurement drop rate again. In comparison to that, the GCEM and the GCB methods deliver a perfect result with zero drops. Thus, the detailed plots of the position and yaw angle errors in Figures 4.36 and 4.37 don't show any surprises.

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.31	1.69	1698	5.64	143.97	1.26
BHAT	0.28	1.68	1871	12.82	143.60	1.31
GCB	0.34	1.69	2616	0.00	143.60	1.23
GCEM	0.35	1.70	2053	0.00	143.60	1.24
BoxFit	0.26	1.91	840	38.97	90.28	

Table 4.12: Evaluated quantities of the scenario Nr. vi in Table 4.4. The BoxFit is obviously well parameterized for this kind of scenario. Only the measurement drop rate brings up another winner. The GCEM and GCB gain a drop rate of 0%

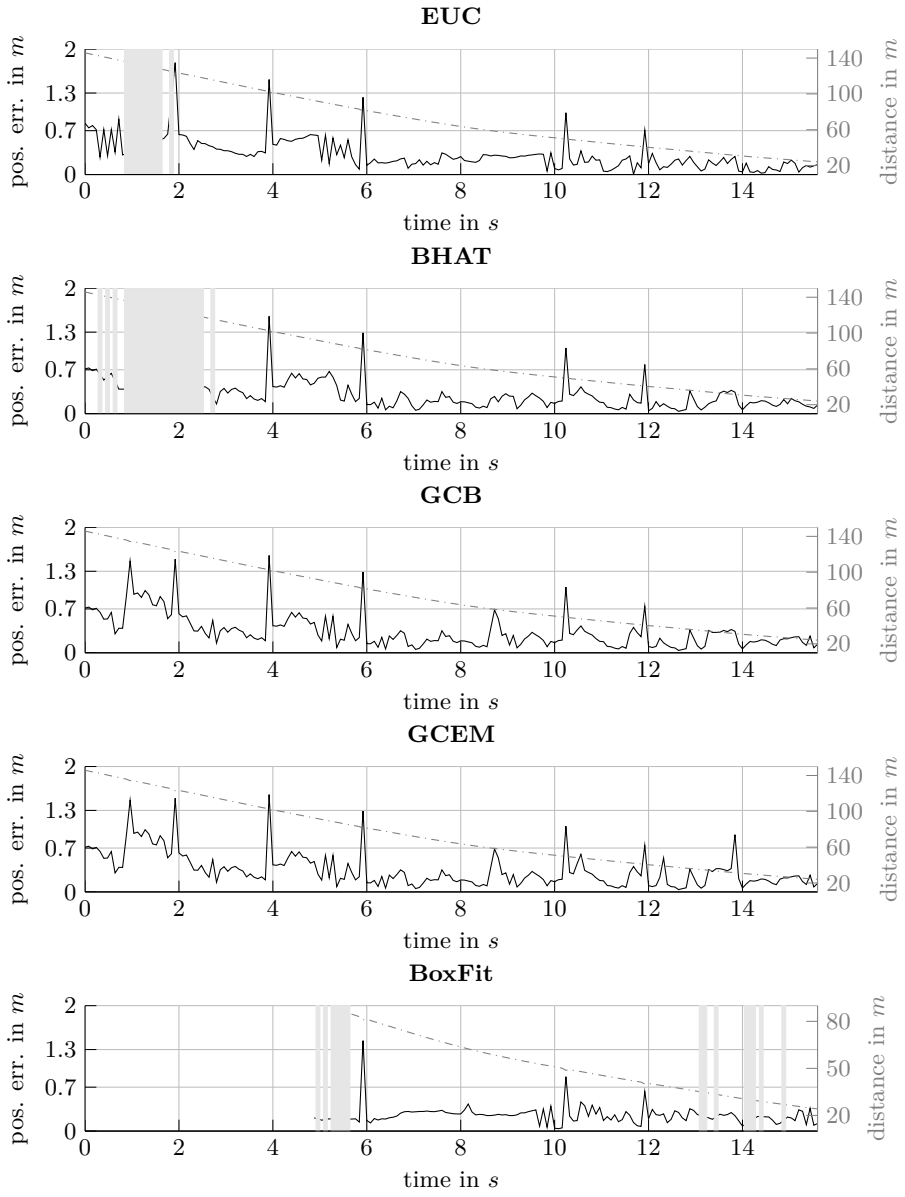


Figure 4.36: Position error over time for scenario Nr. vi in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

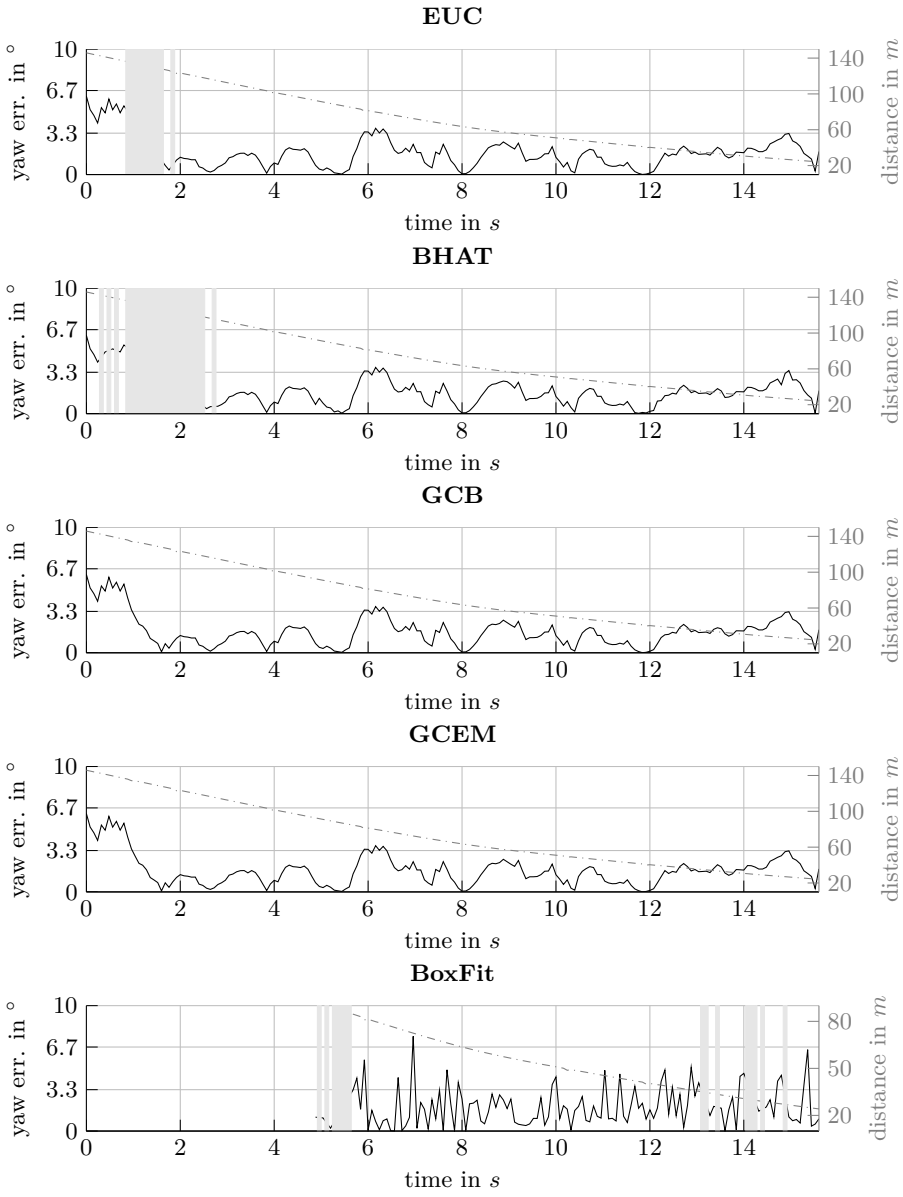


Figure 4.37: Angle error over time for scenario Nr. vi in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Scenario Nr. vii, Follwing while Receding

As Figure 4.38 shows, the perceived vehicle recedes from the ego vehicle while this follows. The position of both vehicles is given in Cartesian coordinates and the time

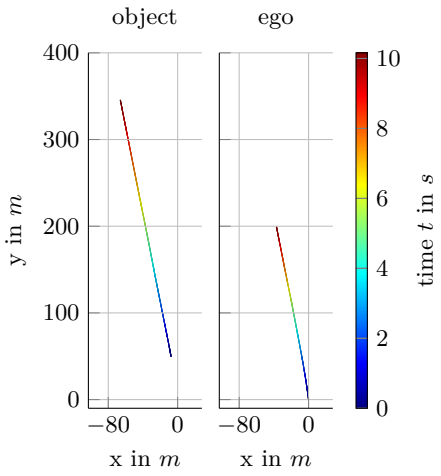


Figure 4.38: Reference and ego trajectory in Cartesian coordinates in the scenario Nr. vii in Table 4.4.

is color coded to get the relation between both trajectories. Like other scenarios already showed, the evaluation quantities in Table 4.13 reveal that the BoxFit is doing very well in rather simple scenarios. The position and angle errors are more than acceptable and the amount of false positives is low. It is the maximum perceived distance and the measurement drops which bring the advantage for the raw data tracking. The errors depicted in Figures 4.39 and 4.40 bring out the problems of the fitting: the shorter perception distance and the measurement drops. Further, this scenario highlights another drawback of the BoxFit: the mean value of the yaw angle error seems to be very good, but taking a closer look at Figure 4.40 shows, that the yaw angle error is very jumpy. The errors are getting very high, but they also can get very low. Expressed as a mean value this information gets lost. It is a bit similar for the GCEM, but since the yaw angle is already filtered, the output is much smoother and there are less peaks.

	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.36	0.84	1440	9.02	148.36	0.62
BHAT	0.36	0.85	1778	9.02	148.29	0.62
GCB	0.37	0.85	2724	3.76	149.33	0.62
GCEM	0.37	0.85	1640	3.76	149.33	0.62
BoxFit	0.22	1.51	572	45.11	94.50	

Table 4.13: Evaluated quantities of the scenario Nr. vii in Table 4.4.

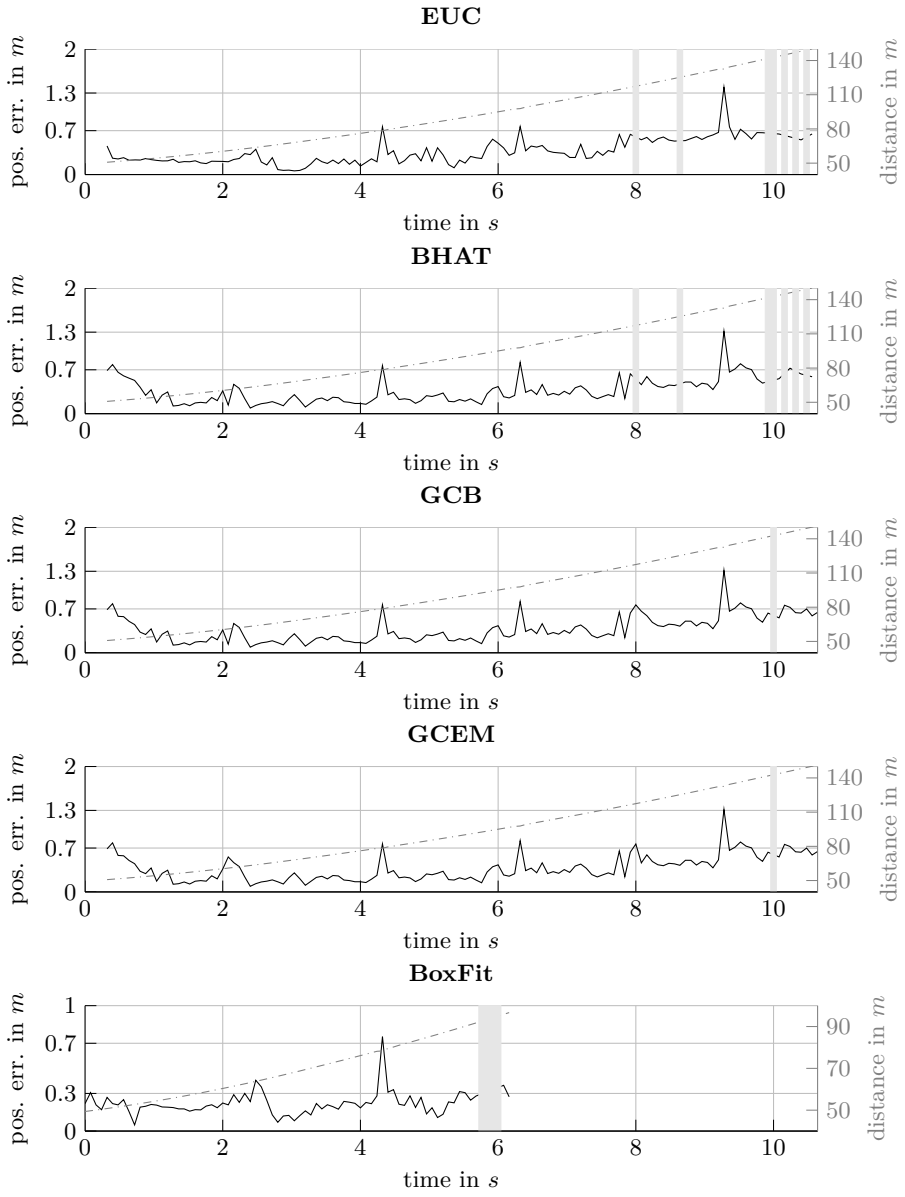


Figure 4.39: Position error over time for scenario Nr. vii in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

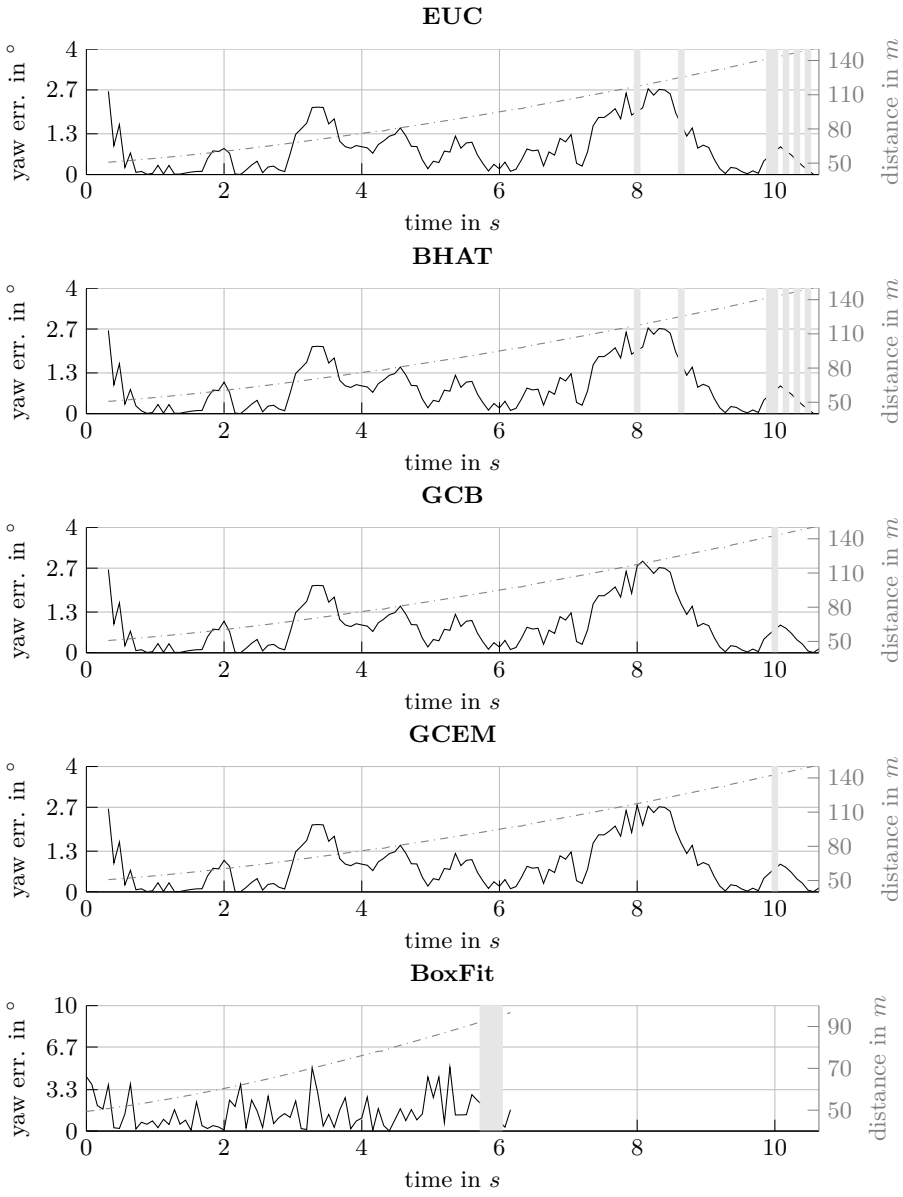


Figure 4.40: Angle error over time for scenario Nr. vii in Table 4.4. The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time period no hypothesis was generated.

Summary

To sum up the results over all evaluated vehicle perception scenarios (Nr. i to Nr. vii, Table 4.4), Table 4.14 shows the mean values of the chosen evaluation quantities. The mean is equivalent to a weighted sum of all sequences, where the weights are

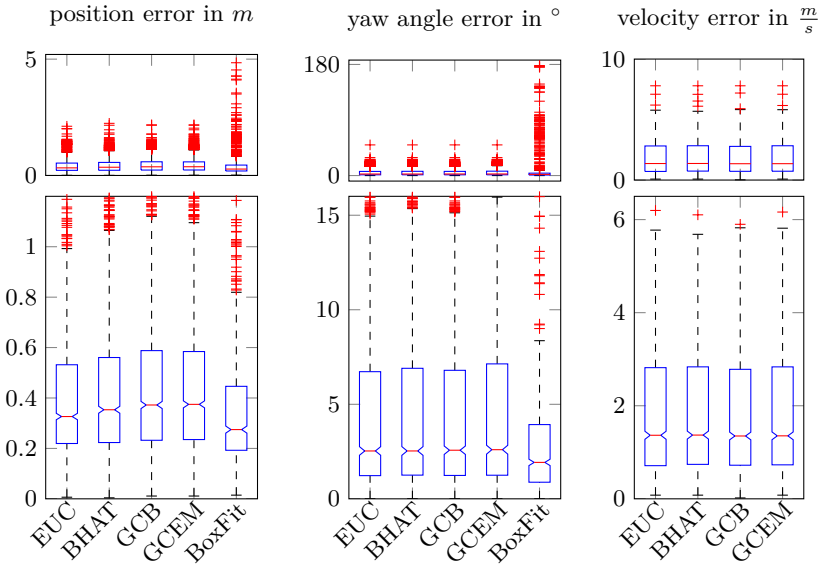
	pos. err. in m	yaw err. in $^\circ$	false positives	drop in %	max. dist. in m	vel. err. in $\frac{m}{s}$
EUC	0.41	4.92	11776	13.90	148.36	1.77
BHAT	0.43	4.90	18405	11.20	148.29	1.78
GCB	0.45	4.92	24599	6.72	149.33	1.76
GCEM	0.45	5.01	16575	5.36	149.33	1.78
BoxFit	0.52	15.71	9082	51.64	107.52	

Table 4.14: Summary of the evaluation over all vehicle perception scenarios (Nr. i to Nr. vii) in Table 4.4. The evaluated quantities are explained in Table 4.6.

proportional to the length of the sequence. The results show, that the position estimation of all compared methods have approximately the same error. Due to the limited accuracy of the reference system mentioned above, these differences are not very meaningful. Since the position is evaluated at the corners of an object, the position error also depends on the measurement point density. Here, the BoxFit yields nearly the same mean as the other methods, and taking a closer look to the boxplot in Figure 4.41a shows that the median error of the BoxFit is even better than those of the other methods. This is mainly due to the fact, that the BoxFit has a very high drop rate and thus the accuracy is only calculated in the easier parts of the scenarios. Additionally, the outliers are widely spread.

The yaw angle error of the presented methods with approximately five degrees is quite good comparing to [RKDW16], where the yaw angle of a vehicle was estimated using high resolution Doppler radar information. Especially regarding the fact, that no object specific model was used. The almost sixteen degrees of the BoxFit are not incapable to be used in subsequent algorithms, but Figure 4.41b reveals the problem of many and widely spread outliers.

As already noted above, the amount of FPs of the BoxFit algorithm is much less than of the other methods. This advantage gets lost with a more strict hypotheses



(a) Boxplot of the position errors.

(b) Boxplot of the yaw angle errors.

(c) Boxplot of the velocity errors.

Figure 4.41: Boxplots of the errors for position, yaw angle, and velocity. All vehicle perception scenarios (Nr. i to Nr. vii) in Table 4.4 were used for this evaluation. Every plot consists of a total view in the upper plot and a close up in the lower plot.

filtering as presented in Section 4.4.3. Even the maximum distance of the BoxFit seems not too bad, but reminding the evaluation in Figures 4.33 and 4.34, a robust detection is only available in up to approximately $80m$ depending on the scenario. Further, the measurement drop rate of the BoxFit is very poor. Additionally, because of the lack of a velocity estimation, the BoxFit can not differ between oncoming and receding objects.

Altogether, the presented approaches outperform the BoxFit used in this work and probably would do so with many clustering and fitting approaches. The four presented methods yield very similar results and mostly differ in the measurement drop rate and the amount of false positives. Since most tracking applications can handle false positives quite effectively but have problems with missed detections the priority is the drop rate. Thus the two clustering methods GCEM and GCB have a small advantage. The GCB method has a higher amount of false positives than the GCEM and has a slightly higher drop rate. Using this priority the GCEM method should be preferred. Having other or more requirements, the recommendation could be the other way round.

4.5.2 Evaluation of the Pedestrian & Bicycle Perception

The purpose of the raw data tracking was a model free approach to allow the perception of vehicles, pedestrians, and others with the same preprocessing algorithm. Although different parameterizations for vehicles and pedestrians are useful, but not mandatory, it is possible to generate hypotheses for both of them. Here, the algorithm is evaluated using three pedestrians and two bicycles after one another. Figure 4.42 shows the trajectories of the five objects. These five objects are temporally separated.

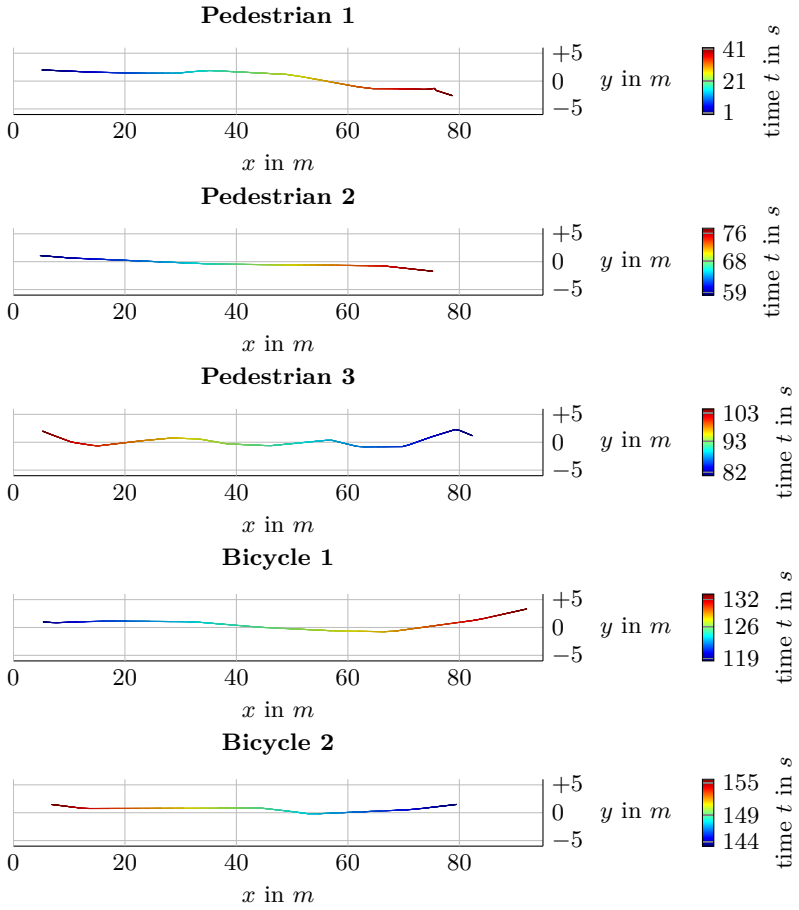


Figure 4.42: Trajectories of the pedestrian & bicycle scenario (Nr. viii in Table 4.4). One long sequence was divided into five parts. The positions are given in vehicle coordinates.

There is always only one object in front of the ego vehicle. This is also visualized using the colored time bar on the right of every trajectory. The chosen surrounding is quite challenging: the observed footpath with a width of about $4m$, as shown in Figure 4.43, is surrounded by dense bushes and trees. Hedges cover both sides of the path. In some segments branches extend into the path. Even with a still standing ego object it is quite difficult to separate the bicycles and pedestrians from the background. This leads to a huge amount of dense data. Furthermore, the leaves

Figure 4.43: Image of the setting the pedestrian and bicycle scenario was taken.

and branches are moving in the wind causing moving raw data tracks. The observed pedestrians and bicycles approach and recede along the vegetation on both sides to test the limitations of the system. As already pointed out, the ground truth was generated manually in a labeling process using the sensor raw data. This is an error prone process but using GNSS based solutions fail in such a narrow surrounding. The evaluation results are depicted in Table 4.15. Since the BoxFit is not able to perceive anything else than vehicles, it is not compared here. The results show that the quite challenging scenario leads to a rather high measurement drop rate. One reason is that there are a lot of moving branches from trees etc. which cause raw data tracks to be moving. This movement has a random direction. If a true object is getting close to the trees and bushes, the tracks on the object might be fused with the tracks on the branches or may be occluded. Another reason seems to be responsible for even more measurement drops: the point cloud on bicycles and pedestrians is very sparse. In a distance of $80m$ it occurs that there is no single point on the object. Figure 4.44 shows, that for the GCB and GCEM method the measurement drops increase drastically with the distance. The position error is again only valid as a trend because of the manual ground truth generation, but it shows

	pos. err. in m	false positives	drop in %	max. dist. in m
EUC	0.81	31773	66.03	84.95
BHAT	0.82	10780	57.50	85.47
GCB	0.83	14523	44.35	89.91
GCEM	0.82	21191	44.81	89.91

Table 4.15: Evaluated quantities of the pedestrian & bicycle scenario (Nr. viii in Table 4.4). The BoxFit is not able to detect other objects than vehicles. Thus, it is not evaluated in this scenario.

that there is hardly any difference between the evaluated methods. The amount of false positives again reflects the difficulty of the chosen scenario. The BHAT method shows less false positives, but also a worse performance. Here, the GCB seems to be the best compromise. Figure 4.44 shows the position error over time in detail. The angle error is not evaluated since the manual labeling of the yaw angle is not reliable. The gray overlays display again the time periods where no hypothesis at the position of the reference was generated. The red boxes are highlighting time periods in which no reference object was moving within the field of view. One can see, that the position error normally increases with the distance and the object is lost in higher distances or if the object gets too close to the surrounding vegetation. It might also come to one's mind that the position error has a static offset. This actually might be true, since the labeling of the position is done in the center of the labeled point cloud and the measurement is given at one of the reference points. This problem was reduced a bit for bicycles since it was possible to approximately label the yaw angle. The results of Table 4.15 and Figure 4.44 show the limitations of the presented approach. Where the GCB and GCEM might perform good enough for subsequent tracking algorithms, the other methods have a way too high drop rate. In higher distances the GCB and GCEM also fail quite often. This is mainly due to the limited resolution of the sensor. As depicted in Figure 4.45 the amount of points on a pedestrian in over $60m$ is zero in the worst case. The alignment of the sensor was chosen to look straight ahead with one layer, so it is likely to have one to two measurement points on the target in high distances. This reflects the results in Figure 4.44, where the performance decreases drastically in distances over $60m$. Regarding this, it is a quite satisfying result to have hypotheses in up to over $80m$. The drops in closer distances are due to track loss or because of too big clusters. This happens if the bushes etc. move in the wind and cause jittering measurement points.

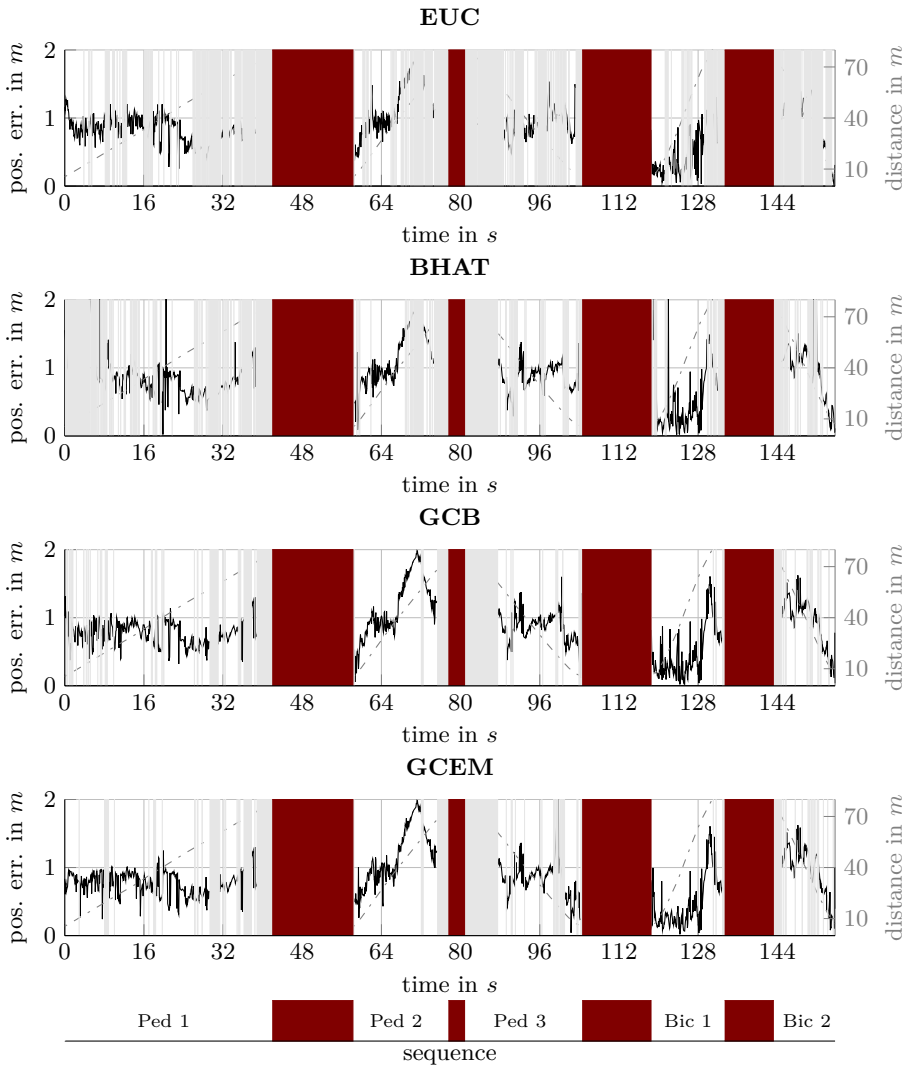


Figure 4.44: Position and yaw angle errors over time for the pedestrian & bicycle scenario (Nr. viii in Table 4.4). The distance to the perceived object is depicted with a gray dash dotted line. The gray overlays display the time where no hypothesis at the position of the reference was generated. The red boxes are the time no reference object was moving within the field of view.

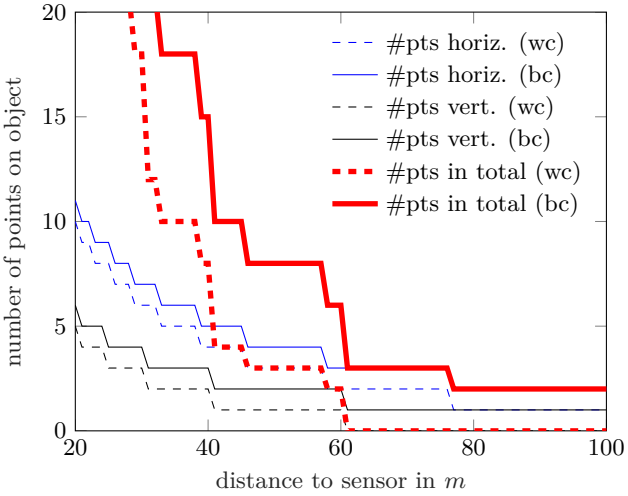


Figure 4.45: Theoretical amount of points from one IBEO LUX 4 on a pedestrian with an assumed width of $0.5m$ and height of $1.7m$. In blue, the number of vertical points (in one layer). In black, the number of horizontal points (over multiple layers). In red the total amount of points on the pedestrian (multiplication of vertical and horizontal). There is always a best case (bc) and a worst case (wc) analysis. The physical effect of beam expansion was not taken into account.

This scenario shows, that an adaption of clustering and hypotheses generation of the algorithm corresponding to the expected type of object might help. Nevertheless, regarding the complexity of the problem and without any special adaption of the algorithm, the performance of the GCB and GCEM are a good result.

4.6 Summary

In this chapter, a new method was presented to generate measurement hypotheses using a high density distance measuring sensor. It does not matter if the sensor measures bearing and range, or Cartesian coordinates. With the new method a linear measurement model can be applied and therefore be used in information filtering without any loss.
The obvious disadvantages are the additional computational load, the poor ability to recognize still standing objects, and the relatively high amount of false positives.

These disadvantages are well compensated for by the following improvements:

- In comparison to simple clustering algorithms like the BoxFit, the perception range is extended markedly as shown in the evaluation in Section 4.5.1.

- The perception is not limited to certain scenarios, shapes, or types of objects. The algorithm is independent from any model assumption.

- The yaw angle estimation is quite confident in every scenario, with the limitation, that filtering the yaw angle in a subsequent tracking application causes a filtering of an already filtered quantity. Since the filtering process is always a type of low pass, double filtering amplifies this behavior.

- A velocity estimation is available. This is very useful for the initialization of new tracks. When using it as a measurement in the subsequent tracking application, the problem of double filtering arises again.

- An estimation of the covariance matrix of the generated hypothesis is available. As explained in Section 4.4.3, the variances are not very precise, but they are a rough estimate and can be used as an upper boundary value.

- Measurements produce an unique ID over long distances which might simplify the task of data association.

- Reducing the amount of false positives using yaw angle and velocity information turned out to be very useful.

Despite all advantages listed above, there are still possibilities for further improvements. First of all, since the algorithm is a proof of concept, the parameterization needs to be optimized based on a comprehensive data set. For better detection of non-moving objects, it seems to be useful to use a threshold on moving objects and implement some sort of fallback solution, e.g. a principle component analysis (PCA), for the non-moving ones. Reducing the amount of false positive measurements would surely be another helpful improvement. As shown above, the yaw angle on its own could be used for this task, but a more extensive classification seems to be promising. Another possibility to improve the hypotheses clusters may be to use symmetric distance measures other than presented in this work. The suggested future work so far was concerning the hypotheses generating algorithm itself, but a massive reduction of the computational load could be achieved by a simple preprocessing of the raw data. The task of such a preprocessing step would be to reduce the amount of data e.g. at the side of the road where only bushes and trees are present. To do this, a sort of attention control is conceivable. Different approaches to such an attention control might be:

- A digital street map can be used to reduce the sensor data to areas, were other road users are expected to be. Further, if the vehicle path is known, the attention could be increased along that path by reducing the resolution in uninteresting regions.

- An online grid map, where cells are recognized as stationary over time. A drawback of this method is that the online grid map is only reliable in an area directly in front of the ego vehicle. Since unnecessary data are mainly caused by exactly this area, the drawback is only a small one. Details of this approach can be found in [NRK+12].

- An offline grid map, where the roads of interest are mapped in advance, extends the functionality of an offline grid map with the disadvantage of invalid maps in cases of changes along the road.

- Simultaneous localization and mapping (SLAM) as a mixture of offline and online grid map. Using SLAM allows localization of the ego vehicle on a formerly created map and incorporating the recent changes along the road simultaneously. Nice summaries on SLAM are given, among others, in [AMV13; BD06; DB06]).

Chapter 5

Conclusion

To handle the increasing complexity of modern sensory systems, like autonomous vehicles, the key is a generic formulation of sensor data fusion systems. The main property a generic fusion system should have is the possibility to exchange sensors easily. The JIPDA is a method which suits the need to fuse information from different types of sources so far. In the first part of this work, it was derived how the information filter can be used within the JIPDA and it was shown that their combination allows the formulation of a totally generic sensor data fusion framework. This framework has multiple advantages: a common sensor interface for all kind of sensors allowing the usage of any sensor without knowledge about the sensor's measurement principle. Additionally, it is not possible to extract detailed knowledge about the sensor from the obtained measurements. This anonymization allows the manufacturer to keep secrecy about the details of the sensor. Finally, the interface has always the same number of values resulting in full exchangeability of sensors without adaption being necessary to the fusion framework. In the course of the work, the new information filter version was shown to be equivalent to the standard Kalman filter approach in linear systems. In the case of non-linear systems, the accuracy strongly depends on the non-linearity and is caused by the use of pseudo-determinants and pseudo-inverses. For non-linear measurement models a linearization at the measurement itself, in case of invertible measurement models, and at the predicted state using a backward channel are feasible. In case of non-invertible measurement models additional assumptions may help to realize a linearization of the measurement model, e.g. the flat world assumption for cameras. Furthermore, it was shown that other fusion systems basing on Gaussian measurement likelihoods can also use the information filter approach to generalize the sensor interface.

Beside a generic fusion framework, the performance of the connected sensor systems is essential to achieve a high precision perception result. At the moment, high density distance measuring sensors delivering dense or semi dense 3D point clouds are very common.

In the second part of this work, a new approach was presented, showing how such point clouds can be processed to get a maximum of information and to suit the needs of a generic framework at the same time. In this new method, the raw sensor data is tracked and the clustering is filtered over time to generate measurement hypotheses. In comparison to model and frame based algorithms, a significant improvement of the perception range and reliability is achieved in many different scenarios. An adaption of the preprocessing to different kinds of objects or scenarios is not necessary any more but still possible. A major advantage of the filtered hypotheses is the availability of velocity and orientation information. Further more, a unique ID and the estimation of the variance of every hypothesis may help subsequent tracking applications during the data association. A detailed evaluation in many different real-data scenarios showed the superiority of the presented raw data tracking and clustering.

The presented approach is a proof of concept and may be further optimized by parameterization and a more comprehensive data set for validation. A possible extension of this work could be to classify the generated hypotheses. Having already information about the general behavior of an object and the corresponding raw data tracks, like the velocity and a possible extent, may be sufficient for this task. The realization of an attention control mechanism or the like, using digital street maps, offline or online grid maps, or a SLAM approach, to improve the results and to reduce false positives is also conceivable.

Acronyms

ADAS	advanced driver assistance systems	ix
ADMA	automotive dynamic motion analyzer	77
BC	birth candidate	15
BD	Bhattacharyya distance	65
CC	center center (reference point position)	70, 71
CL	center left (reference point position)	70, 71
CMKF	converted measurement Kalman filter	31
CR	center right (reference point position)	70, 71
CRLB	Cramer-Rao lower bound	7
CV	constant velocity	39
CVCY	constant velocity and constant turn	34
DBSCAN	density-based spatial clustering of applications with noise	44
EIF	extended information filter	8
EKF	extended Kalman filter	5
FC	front center (reference point position)	70, 71
FISST	finite set statistics	xi
FL	front left (reference point position)	70
FN	false negative	15
FP	false positive	15
FR	front right (reference point position)	70
GC	graph clustering	62
GNN	global nearest neighbor	10
GNSS	global navigation satellite system	77
HDDM	high density distance measuring	43

ID	identifier	10
IEPF	iterative end point fit	44
IMU	inertial measurement unit	77
IPDA	integrated probabilistic data association	59
JIPDA	joint integrated probabilistic data association	1
JPDA	joint probabilistic data association	14
LiDAR	light detection and ranging	39
LNN	local nearest neighbor	10
LUT	look up table	15
MHD	Mahalanobis distance	10
MHT	multi-hypothesis tracking	54
NN	nearest neighbor	10
OMAT	optimal mass transfer	18
OSPA	optimal subpattern assignment	19
OSPAT	optimal subpattern assignment for tracks	19
PCA	principle component analysis	116
PDA	probabilistic data association	3
PHD	probability hypothesis density	46
RC	rear center (reference point position)	70, 71
RFS	random finite set	46
RL	rear left (reference point position)	70
RMSE	root mean square error	80
RR	rear right (reference point position)	70
SLAM	simultaneous localization and mapping	117
SMC	sequential Monte Carlo	33
TN	true negative	15
TP	true positive	15
UIF	unscented information filter	8
UKF	unscented Kalman filter	5
UTM	universal transverse mercator	79

List of Symbols

Notations

$\mathrm{diag}(a, b, \dots)$	diagonal matrix with elements $[a, b, \dots]$ on the main diagonal
$\delta_l(\cdot)$	Dirac delta function
$\hat{\cdot}$	estimated value
$\mathrm{E}\{\cdot\}$	expectation
$\Theta(\cdot)$	Heaviside function
$\cdot^{-1}$	inverse of value or matrix
$\underline{\cdot}$	labeled value, mostly used for labeled state vectors as $\underline{\boldsymbol{x}}$
$\bar{\cdot}$	mean value
∇	partial derivative operator (nabla)
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	normal distribution with n dimensional mean $\boldsymbol{\mu}$ and the corresponding covariance $\boldsymbol{\Sigma}$
$p(a\|b)$	conditional probability distribution for a given b
$p(a_1, a_2, \dots, a_n)$	probability distribution for n random variables
$\cdot^{\dagger}$	Moore-Penrose pseudo-inverse of a matrix
$\|\cdot\|_+$	pseudo-determinant of a matrix
$\mathrm{rank}(\cdot)$	rank of a matrix
$\boldsymbol{R}(\varrho)$	two dimensional rotation matrix for rotation by angle ϱ
$\boldsymbol{s}(\cdot)$	score function
$\cdot^{\mathrm{T}}$	transpose of vector or matrix

Specific probabilities

p_D	detection probability
p_G	gating probability
$p(e)$	edge likelihood in graph based JIPDA
p_P	persistence probability
$p_\exists$	existence probability
$p_{\textcircled{b},prior}$	a priori birth probability

p_F	sensory clutter probability
$p_{ⓑ}$	spatial birth probability
$p_{\ TP}$	true positive probability

Subscripts and superscripts

$\{0, +2\pi, -2\pi\}$	superscript to indicate that 0, $+2\pi$, or -2π was added to the yaw angle of the state
j	measurement index
k	time step
$k\|k$	estimated and predicted value for time step k
$k\|k-1$	estimated for time step $k-1$ and predicted for time step k
t	tracked object index

General symbols

b	clutter model
β	association weight
ⓑ	symbol for birth candidate in graph based JIPDA
ⓒ	clutter symbol in graph based JIPDA
c_n	n dimensional unity sphere
$dim(\boldsymbol{z})$	dimension of the measurement vector
$\boldsymbol{\eta}$	equivalent of the measurement residuum in information space
$\boldsymbol{\mathcal{I}}$	Fisher information
$\boldsymbol{F}$	state transition matrix
γ	constant parameter which can be obtained from the quantile tables chi-square distribution for constant p_G
$\boldsymbol{G}$	control input matrix
$\boldsymbol{H}$	measurement matrix
$\boldsymbol{i}$	information gain
$\boldsymbol{I}$	information gain matrix
$\boldsymbol{K}$	Kalmain gain
Λ	a priori measurement likelihood
$\lambda_{ⓒ}$	spatial densitiy of false measurements
$d_{j,k}^{MHD}$	Mahalanobis distance
$\varnothing$	symbol for missed detection in graph based JIPDA
m	number of measurements
n_z	dimension of the measurement space
$\boldsymbol{\nu}$	measurement residuum

$\mathfrak{a}$	mismatching labels penalization parameter of the OSPA and OSPAT
$\mathfrak{c}$	cut-off parameter of the OSPA and OSPAT
$\mathfrak{p}$	outlier penalization parameter of the OSPA and OSPAT
$d_{\mathfrak{c}}^{\mathfrak{p}}$	OSPA distance for sets
$\boldsymbol{P}$	state covariance matrix
Π	set of possible permutations
$\boldsymbol{Q}$	process noise matrix
$\boldsymbol{R}$	measurement covariance matrix
d_{grid}	size of a grid cell of the discretization grid of preprocessing
$\mathcal{H}$	hypothesis generated by the preprocessing
$l_{\mathcal{H}}$	extent(length) of an hypothesis $\mathcal{H}$
$w_{\mathcal{H}}$	extent(width) of an hypothesis $\mathcal{H}$
$\mathfrak{h}$	hypothesis state vector $\mathfrak{h} = [x_{\mathcal{H}}, y_{\mathcal{H}}, v_{\mathcal{H}}, \psi_{\mathcal{H}}]^{\mathrm{T}}$
$\psi_{\mathcal{H}}$	yaw angle (orientation) of a hypothesis state vector $\mathfrak{h}$
$v_{\mathcal{H}}$	velocity of a hypothesis state vector $\mathfrak{h}$
$x_{\mathcal{H}}$	position in x direction of a hypothesis state vector $\mathfrak{h}$
$y_{\mathcal{H}}$	position in y direction of a hypothesis state vector $\mathfrak{h}$
$\boldsymbol{P}_{\mathcal{H}}$	covariance matrix of an hypothesis state vector $\mathfrak{h}$
$\mathcal{RP}_{\mathcal{H}}$	given reference point for a generated hypothesis $\mathcal{H}$
$\boldsymbol{S}$	innovation covariance matrix
t	tracked object
$\boldsymbol{u}$	control vector
$\boldsymbol{v}$	discrete zero-mean white Gaussian process noise
$\boldsymbol{\Upsilon}$	equivalent of the innovation covariance matrix in information space
V	volume of the elliptical validation (gating) region
$\boldsymbol{w}$	discrete zero-mean white Gaussian measurement noise
d_{BD}	Bhattacharyya distance
$[x_{LS}\ y_{LS}]^{\mathrm{T}}$	measurement point of a laser scanner in Cartesian coordinates
X	set of states
$\boldsymbol{x}$	state vector
ψ	yaw angle
ω	yaw rate
$\boldsymbol{Y}$	information matrix
$\boldsymbol{y}$	information vector
$\boldsymbol{z}$	measurement vector
Z	set of measurements

Bibliography

[AMV13] M. Adams; J. Mullane; and B.-N. Vo: *Circumventing the Feature Association Problem in SLAM.* In: *IEEE Intelligent Transportation Systems Magazin*, volume 5, no. 3, pages 40–58, 2013.

[BD06] T. Bailey and H. Durrant-Whyte: *Simultaneous localization and mapping (SLAM): Part II.* In: *IEEE Robotics & Automation Magazine*, volume 13, no. 3, pages 108–117, 2006.

[Ber88] D. P. Bertsekas: *The auction algorithm: A distributed relaxation method for the assignment problem.* In: *Annals of Operations Research*, volume 14, no. 1, pages 105–123, 1988.

[BF88] Y. Bar-Shalom and T. Fortmann: *Tracking and Data Association.* Academic Press, Inc., 1988.

[Bha46] A. Bhattacharyya: *On a Measure of Divergence between Two Multinomial Populations.* In: *The Indian Journal of Statistics*, volume 7, no. 4, pages 401–406, 1946.

[BL93] Y. Bar-Shalom and X.-R. Li: *Estimation and Tracking: Principles, Techniques, and Software.* Artech House Inc., 1993.

[BL95] Y. Bar-Shalom and X.-R. Li: *Multitarget-Multisensor Tracking: Principles and Techniques.* Volume 3. YBS Publishing, 1995.

[BL98] Y. Bar-Shalom and X.-R. Li: *Estimation and Tracking: Principles, Techniques, and Software.* YBS Publishing, 1998.

[BT75] Y. Bar-Shalom and E. Tse: *Tracking in a cluttered environment with probabilistic data association.* In: *Automatica*, volume 11, no. 5, pages 451–460, 1975.

[DB06] H. Durrant-Whyte and T. Bailey: *Simultaneous localization and mapping: part I.* In: *IEEE Robotics & Automation Magazine*, volume 13, no. 2, pages 99–110, 2006.

[DB93] J. Dezert and Y. Bar-Shalom: *Joint probabilistic data association for autonomous navigation.* In: *IEEE Transactions on Aerospace and Electronic Systems*, volume 29, no. 4, pages 1275–1286, 1993.

[DHS01] R. O. Duda; P. E. Hart; and D. G. Stork: *Pattern Classification.* Volume 2. John Wiley & Sons, Inc., 2001.

[Dur01] H. Durrant-Whyte: *Multi Sensor Data Fusion*. 2001.

[EAGG13] V. Edman; M. Andersson; K. Granström; and F. Gustafsson: *Pedestrian Group Tracking Using the GM-PHD Filter*. In: *Proceedings of the European Signal Processing Conference*, Marrakech, Morocco, 2013.

[EKSX96] M. Ester; H.-P. Kriegel; J. Sander; and X. Xu: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*. In: *International Conference on Knowledge Discovery and Data Mining*, pages 22–231, 1996.

[FD93] M. Fernandez and H. Durrant-Whyte: *An Information-Theoretic Approach to Data-Validation*. In: *Proceedings of the American Control Conference*, pages 2351–2355, 1993.

[FF08] M. Feldmann and D. Franken: *Tracking of extended objects and group targets using random matrices - a new approach*. In: *Proceedings of the 11th International Conference on Information Fusion*, pages 1–8, 2008.

[GDH92] S. Grime; H. F. Durrant-Whyte; and P. Ho: *Communication in Decentralized Data-Fusion Systems*. In: *Proceedings of the American Control Conference*, pages 3299–3303, 1992.

[Gen09] GeneSys Elektronik GmbH: *Bedienungsanleitung ADMA*. Tech. rep. 2x.6.1.4. Bedienungsanleitung, GeneSys Elektronik GmbH, Offenburg, 2009.

[GLO10] K. Granström; C. Lundquist; and U. Orguner: *A Gaussian Mixture PHD Filter for Extended Target Tracking*. In: *Proceedings of the 13th International Conference on Information Fusion*, pages 1–8, 2010.

[Gri10] M. Grinberg: *Data Association for Multi-Target-Tracking*. Tech. rep. IES-2010-12. Karlsruhe Institute of Technology, 2010, pages 165–186.

[HL64] Y.-C. Ho and R. Lee: *A Bayesian approach to problems in stochastic estimation and control*. In: *IEEE Transactions on Automatic Control*, volume 9, no. 4, pages 333–339, 1964.

[HM04] J. Hoffman and R. Mahler: *Multitarget miss distance via optimal assignment*. In: *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, volume 34, no. 3, pages 327–336, 2004.

[HZ04] R. I. Hartley and A. Zisserman: *Multiple View Geometry in Computer Vision*. Second, Cambridge University Press, ISBN: 0521540518, 2004.

[Ibe13] Ibeo Automotive Systems GmbH: *LUX 4*. Data Sheet. Ibeo Automotive Systems GmbH, 2013.

[JUD00] S. Julier; J. Uhlmann; and H. Durrant-Whyte: *A new method for the nonlinear transformation of means and covariances in filters and estimators*. In: *IEEE Transactions on Automatic Control*, volume 45, no. 3, pages 477–482, 2000.

[Kal60] R. E. Kalman: *A New Approach to Linear Filtering and Prediction Problems*. In: *Transactions of the ASME - Journal of Basic Engineering*, volume 82, pages 35–45, 1960.

[Käm07] N. Kämpchen: *Feature-level fusion of laser scanner and video data for advanced driver assistance systems*. PhD thesis, Ulm University, 2007.

[KNW+15] F. Kunz; D. Nuss; J. Wiest; et al.: *Autonomous driving at Ulm University: A modular, robust, and sensor-independent fusion approach*. In: *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 666–673, 2015.

[Kuh55] H. W. Kuhn: *The Hungarian Method for the Assignment Problem*. In: *Naval Research Logistics Quarterly*, volume 2, pages 83–97, 1955.

[Lee08] D.-J. Lee: *Nonlinear Estimation and Multiple Sensor Fusion Using Unscented Information Filtering*. In: *IEEE Signal Processing Letters*, volume 15, pages 861–864, 2008.

[MA97] A. G. O. Mutambara and M. Al-Haik: *State and information space estimation: a comparison*. In: *Proceedings of the American Control Conference*. Volume 4, pages 2374–2375, 1997.

[Mah03] R. Mahler: *Multitarget Bayes filtering via first-order multitarget moments*. In: *IEEE Transactions on Aerospace and Electronic Systems*, volume 39, no. 4, pages 1152–1178, 2003.

[Mäh09] M. Mählisch: *Filtersynthese zur simultanen Minimierung von Existenz-, Assoziations- und Zustandsunsicherheiten in der Fahrzeugumfelderfassung mit heterogenen Sensordaten*. PhD thesis, Ulm University, 2009.

[MDM09] M. Munz; K. Dietmayer; and M. Mählisch: *A Sensor Independent Probabilistic Fusion System for Driver Assistance Systems*. In: *Proceedings of the 12th International Conference on Intelligent Transportation Systems*, 2009.

[ME02] D. Musicki and R. Evans: *Joint Integrated Probabilistic Data Association - JIPDA*. In: *Proceedings of the 5th International Conference on Information Fusion*. Volume 2, pages 1120–1125, 2002.

[Mei16] D. Meissner: *Intersection-based road user tracking using a classifying multiple-model PHD filter*. PhD thesis, Ulm University, 2016.

[MES94] D. Musicki; R. Evans; and S. Stankovic: *Integrated probabilistic data association*. In: *IEEE Transactions on Automatic Control*, volume 39, no. 6, pages 1237–1241, 1994.

[Min98] T. P. Minka: *Inferring a Gaussian distribution*. Tech. rep. 1998.

[MM04] D. Musicki and M. R. Morelande: *Gate Volume Estimation for Target Tracking*. In: *Proceedings of the 7th International Conference on Information Fusion*, 2004.

[Moo20] E. H. Moore: *On the reciprocal of the general algebraic matrix*. In: . *Bulletin of the American Mathematical Society*, volume 26, pages 385–396, 1920.

[MRD13] D. Meissner; S. Reuter; and K. Dietmayer: *Combining the 2D and 3D World: A New Approach for Point Cloud Based Object Detection*. In: *Proceedings of the Intelligent Signal Processing Conference*, 2013.

[MSMD08] M. Mählisch; M. Szczot; M. Munz; and K. Dietmayer: *Simultaneous Processing of Multitarget State Measurements and Object Individual Sensory Existence Evidence with the Joint Integrated Probabilistic Data Association Filter*. In: *Proceedings of 5th International Workshop on Intelligent Transportation, Hamburg, Germany*, pages 1–6, 2008.

[Mun11] M. Munz: *Generisches Sensorfusionsframework zur gleichzeitigen Zustands- und Existenzschätzung für die Fahrzeugumfelderkennung*. PhD thesis, Ulm University, 2011.

[Mur68] K. Murty: *An Algorithm for Ranking all the Assignments in Order of Increasing Cost*. In: *Operations Research*, volume 16, pages 682–687, 1968.

[Mut98] A. G. Mutambara: *Decentralized Estimation and Control for Multisensor Systems*. CRC Press, 1998.

[NRK+12] D. Nuss; S. Reuter; M. Konrad; M. Munz; and K. Dietmayer: *Using grid maps to reduce the number of false positive measurements in advanced driver assistance systems*. In: *Proceedings of the 15th International Conference on Intelligent Transportation Systems*, pages 1509–1514, 2012.

[NSD14] D. Nuss; M. Stuebler; and K. Dietmayer: *Consistent environmental modeling by use of occupancy grid maps, digital road maps, and multi-object tracking*. In: *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 1371–1377, 2014.

[Ram72] U. Ramer: *An iterative procedure for the polygonal approximation of plane curves*. In: *Computer Graphics and Image Processing*, volume 1, no. 3, pages 244–256, 1972.

[RD09] S. Reuter and K. Dietmayer: *Fuzzy Estimation and Segmentation for Laser Range Scans*. In: *Proceedings of the 12th International Conference on Information Fusion*, pages 1974–1981, 2009.

[RKDW16] F. Roos; D. Kellner; J. Dickmann; and C. Waldschmidt: *Reliable Orientation Estimation of Vehicles in High-Resolution Radar Images*. In: *Transactions on Microwave Theory and Techniques*, volume 64, no. 9, pages 2986–2993, 2016.

[RVCV11] B. Ristic; B.-N. Vo; D. Clark; and B.-T. Vo: *A Metric for Performance Evaluation of Multi-Target Tracking Algorithms*. In: *IEEE Transactions on Signal Processing*, volume 59, no. 7, pages 3452–3457, 2011.

[SC12] A. Swain and D. Clark: *The PHD filter for extended target tracking with estimable extent shape parameters of varying size*. In: *Proceedings of the 15th International Conference on Information Fusion*, pages 1111–1118, 2012.

[SRD16] A. Scheel; S. Reuter; and K. Dietmayer: *Using separable likelihoods for laser-based vehicle tracking with a Labeled Multi-Bernoulli filter*. In: *Proceedings of the 19th International Conference on Information Fusion*, pages 1200–1207, 2016.

[SS06] N. Saunier and T. Sayed: *A feature-based tracking algorithm for vehicles in intersections*. In: *The 3rd Canadian Conference on Computer and Robot Vision*, pages 59–66, 2006.

[Sto10] B. Storrer: *Entwicklung und Untersuchung von Existenzmodellen für Laserscanner*. MA thesis, Ulm University, 2010.

[SVV08] D. Schuhmacher; B.-T. Vo; and B.-N. Vo: *A Consistent Metric for Performance Evaluation of Multi-Object Filters*. In: *IEEE Transactions on Signal Processing*, volume 56, no. 8, pages 3447–3457, 2008.

[SWBH12] K. Schueler; T. Weiherer; E. Bouzouraa; and U. Hofmann: *360 Degree multi sensor fusion for static and dynamic obstacles*. In: *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 692–697, 2012.

[TBF05] S. Thrun; W. Burgard; and D. Fox: *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[VW05] T. Vercauteren and X. Wang: *Decentralized sigma-point information filters for target tracking in collaborative sensor networks*. In: *IEEE Transactions on Signal Processing*, volume 53, no. 8, pages 2997–3009, 2005.

[Wik17] Wikipedia: *Waymo — Wikipedia, The Free Encyclopedia*. [Online; accessed 9-January-2017]. 2017. URL: `https://en.wikipedia.org/w/index.php?title=Waymo&oldid=759068187`.

[YNK+15] T. Yuan; D. S. Nuss; G. Krehl; M. Maile; and A. Gern: *Fundamental properties of dynamic occupancy grid systems for vehicle environment perception*. In: *6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pages 153–156, 2015.

Publications

Parts of this thesis were pre-released in the author's journal articles and conference papers.

D. Meissner; S. Reuter; B. Wilking; and K. Dietmayer: *Road User Tracking Using a Dempster-Shafer Based Classifying Multiple-Model PHD Filter*. In: *Proceedings of the 16th International Conference on Information Fusion*, pages 1236–1242, 2013.

D. Nuss; B. Wilking; J. Wiest; et al.: *Decision-Free True Positive Estimation with Grid Maps for Multi-Object Tracking*. In: *Proceedings of the 16th International Conference on Intelligent Transportation Systems*, pages 28–34, The Hague, Netherlands, 2013.

S. Reuter; D. Meissner; B. Wilking; and K. Dietmayer: *Cardinality Balanced Multi-Target Multi-Bernoulli Filtering Using Adaptive Birth Distributions*. In: *Proceedings of the 16th International Conference on Information Fusion*, pages 1608–1615, 2013.

S. Reuter; B.-T. Vo; B. Wilking; D. Meissner; and K. Dietmayer: *Divergence Detectors for the δ-Generalized Labeled Multi-Bernoulli Filter*. In: *Workshop on Sensor Data Fusion: Trends, Solutions, Applications*, Bonn, Germany, 2013.

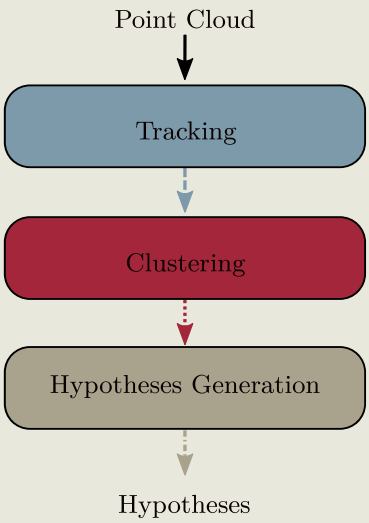
S. Reuter; B. Wilking; and K. Dietmayer: *Methods to Model the Motion of Extended Objects in Multi-Object Bayes Filters*. In: *Proceedings of the 15th International Conference on Information Fusion*, pages 527–534, 2012.

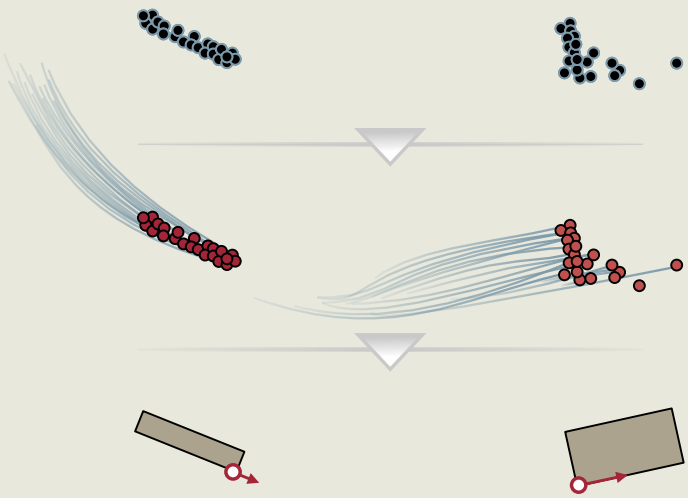
S. Reuter; B. Wilking; J. Wiest; M. Munz; and K. Dietmayer: *Real-Time Multi-Object Tracking using Random Finite Sets*. In: *Transactions on Aerospace and Electronic Systems*, volume 49, no. 4. to appear, pages 2666–2678, 2013.

E. Strigel; D. Meissner; F. Seeliger; B. Wilking; and K. Dietmayer: *The Ko-PER intersection laserscanner and video dataset*. In: *Proceedings of the 17th International Conference on Intelligent Transportation Systems*, pages 1900–1901, 2014.

B. Wilking; D. Meissner; S. Reuter; and K. Dietmayer: *Information Maps: A Practical Approach to Position Dependent Parameterization*. In: *arXiv:1312.3808v1 [cs.CE]*, 2013.

B. Wilking; S. Reuter; and K. Dietmayer: *Probabilistic Data Association in Information Space for Generic Sensor Data Fusion*. In: *Proceedings of the 15th International Conference on Information Fusion*, pages 317–323, 2012.





ISBN 978-3-941543-37-9

e-ISBN 978-3-941543-38-6